PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2002-077137**
(43)Date of publication of application : **15.03.2002**

(51)Int.Cl. **H04L 9/14**
G06F 12/14
G09C 1/00

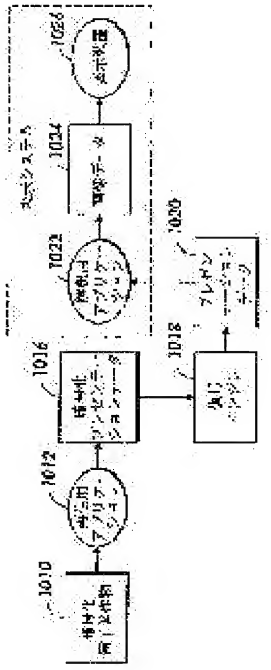
(21)Application number : **2001-087860**
(22)Date of filing : **26.03.2001**

(71)Applicant : **CONTENTGUARD HOLDINGS INC**
(72)Inventor : **WANG XIN**

(30)Priority
Priority number : **2000 536089** Priority date : **24.03.2000** Priority country : **US**

(54) SYSTEM AND METHOD FOR PROTECTION OF DIGITAL WORKS

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a method of protecting a digital work which uses a format preserving encryption scheme to encrypt the digital work.
SOLUTION: This method enables any native replay application 1012 or rendering application 1022 to transform an encrypted digital work 1010 into encrypted presentation data 1016. The originator's digital content is protected in its original form by not being decrypted. This method enables the rendering or replay application 1012 to process the encrypted document into encrypted presentation data 1016 without decrypting it first. Encrypted presentation data is then decrypted just before it is displayed 1026 to the user. An additive encryption scheme is a particular type of encryption scheme which preserves formatting of a digital work.



LEGAL STATUS

[Date of request for examination] 18.12.2002
[Date of sending the examiner's decision of rejection] 18.07.2006
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2002－77137
（P2002－77137A）
(43)公開日　平成14年３月15日(2002.3.15)

(51)Int.Cl.[7]	識別記号	ＦＩ		テーマコード（参考）	
Ｈ０４Ｌ　9/14		Ｃ０６Ｆ　12/14	３２０Ｂ	５Ｂ０１７	
Ｇ０６Ｆ　12/14	３２０	Ｃ０９Ｃ　1/00	６６０Ｄ	５Ｊ１０４	
Ｇ０９Ｃ　1/00	６６０	Ｈ０４Ｌ　9/00	６４１		

審査請求　未請求　請求項の数２　ＯＬ　外国語出願　（全103頁）

(21)出願番号　特願2001－87860(P2001－87860)

(22)出願日　平成13年３月26日(2001.3.26)

(31)優先権主張番号　５３６０８９

(32)優先日　平成12年３月24日(2000.3.24)

(33)優先権主張国　米国（ＵＳ）

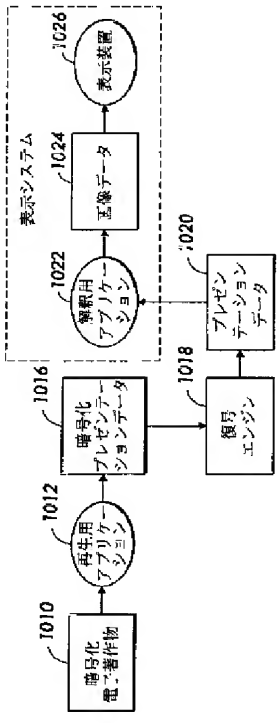
(71)出願人　500470703
コンテントガード　ホールディングズ　インコーポレイテッド
ＣｏｎｔｅｎｔＧｕａｒｄ　Ｈｏｌｄｉｎｇｓ，　Ｉｎｃ．
アメリカ合衆国　19803　デラウェア州　ウィルミントン　スイート　205－エム　フォーク　ロード　103

(74)代理人　100079049
弁理士　中島　淳　（外１名）

最終頁に続く

(54)【発明の名称】　電子著作物の保護方法及び保護システム

(57)【要約】　（修正有）
【課題】電子著作物の暗号化に形式保存暗号化方式を用いる電子著作物保護方法の提供。
【解決手段】この方法により、あらゆる在来の再生用アプリケーション１０１２又は解釈用アプリケーション１０２２は、暗号化された電子著作物１０１０を暗号化プレゼンテーションデータ１０１６に変換することができ、発信者の電子コンテンツは、復号されないことにより元の形式で保護される。この方法により、解釈用又は再生用アプリケーション１０１２は、暗号化文書を初めに復号せずに暗号化プレゼンテーションデータ１０１６に処理することができる。そして、暗号化プレゼンテーションデータは、ユーザに表示１０２６する直前に復号される。この加法的暗号化方式は、電子著作物の形式を保つ特殊なタイプの暗号化方式である。



【特許請求の範囲】
【請求項１】　変換関数Fによるプレゼンテーションデータ F(z)への変換の際に、電子コンテンツ及び形式情報を含む電子著作物zを保護する方法であって、
前記電子著作物zを形式保存暗号化方式Eに従って暗号化するステップと、
暗号化された電子著作物E(z)を暗号化プレゼンテーションデータF(E(z))に変換するステップと、
前記暗号化プレゼンテーションデータF(E(z))を復号関数Dに従って復号し、前記プレゼンテーションデータF(z)を得るステップであって、D(F(E(z))=F(z)である、ステップと、
を有する電子著作物zを保護する方法。
【請求項２】　変換関数Fによるプレゼンテーションデータ F(z)への変換の際に、電子コンテンツ及び形式情報を含む電子著作物zを保護するシステムであって、
前記電子著作物zを形式保存暗号化方式Eに従って暗号化する暗号化エンジンと、
暗号化された電子著作物E(z)を暗号化プレゼンテーションデータF(E(z))に変換する変換関数と、
前記暗号化プレゼンテーションデータF(E(z))を復号関数Dに従って復号し、前記プレゼンテーションデータF(z)を得るための復号エンジンであって、D(F(E(z))=F(z)である、復号エンジンと、
を有する電子著作物zを保護するシステム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、文書に対する権利の管理に係り、特に、盲目的変換を可能にする形式保存暗号化を用いる、電子著作物の保護方法に関する。
【０００２】
【従来の技術】電子商取引を介して電子文書又は著作物が広範囲に普及するのを妨げている重要な問題の１つは、現段階では、これらの電子文書又は著作物の配布及び使用時にコンテンツ所有者の知的所有権の保護が十分でない点である。この問題を解決する試みは、「知的所有権管理」（ＩＰＲＭ）、「デジタル所有権管理」（ＤＰＲＭ）、「知的所有管理」（ＩＰＭ）、「権利管理」（ＲＭ）、「デジタル権利管理」（ＤＲＭ）、及び「電子版権管理」（ＥＣＭ）等と呼ばれている。デジタル権利管理の核心には、許可を受けたユーザのみが、取得した電子文書又は著作物を操作できることを保証する、という根本的な論点がある。コンテンツにアクセスされても、コンテンツ所有者の権利の主張に違反したコンテンツの配布及び使用を行われてはならない。
【０００３】ここで言う文書又は著作物とは配布または譲渡等が行われる、あらゆる単位の情報のことで、通信文書、書籍、雑誌、ジャーナル、新聞、他の書類、ソフトウェア、写真及び他の画像、オーディオ及びビデオクリップ、その他のマルチメディアプレゼンテーション等であるが、これらに限られているわけではない。文書の具体的な形式は、紙に印刷するか、記憶媒体上の電子データか、または各種媒体上に他の既存の形式で記録されたものである。本明細書に用いられる電子著作物とは、デジタル形式で維持され、デバイス又はソフトウェアプログラムを用いた再生又は解釈が可能なあらゆる文書、テキスト、オーディオ、マルチメディア、又は他のタイプの著作物もしくはその一部分である。
【０００４】印刷された文書の場合、著者が作成した著作物は通常は出版業者に渡され、そこで著作物の形式が整えられ大量のコピーが印刷される。これらのコピーは、配送業者により書店または他の小売店に送られ、エンドユーザが購入する。
【０００５】印刷文書の場合、通常はコピーの品質が悪く配布費用も高価であったため、違法コピーは抑止されていた。これに対し、電子文書の場合は保護されていないとコピー、修正、及び再配布がきわめて容易である。したがって、電子文書を保護するための何等かの方法を採用し、違法コピーを簡単に行えないようにする必要がある。このような方法が確立すれば、印刷文書のハードコピーを作成して従来の方法で複写する等は可能であっても、コピーの抑止には役立つと思われる。
【０００６】印刷文書の場合、文書をデジタル化するというステップを踏まないと、電子的に再配布することはできない。この制限は抑止として役立つ。しかし、一般的には、ローカルエリアネットワーク（ＬＡＮ）、イントラネット、及びインターネットを介して接続したパーソナルコンピュータ、ワークステーション、他の装置等の現在の汎用コンピューティング及び通信システムの下では、電子文書を許可を受けずに配布することを防止する有効な方法がないことも現実である。無許可コピーを防止するためにハードウェアを使用して解決する方法も何度か試みられたが、成功したとは言えない。
【０００７】文書保護の問題を解決する試みとして、セキュアコンテナー（暗号化メカニズムに依存するシステム）及び高信頼性システムといった２つの基本的な方式が用いられている。
【０００８】暗号化メカニズムは文書を暗号化する。次に、暗号化された文書はパブリックに配布されて保存され、最終的には許可を受けたユーザによってプライベートに復号される。暗号化メカニズムは、パブリックネットワークを介して文書配布業者から目的ユーザに文書を配送する際と、安全でない媒体に文書を記憶する際に、基本形式の保護を提供する。デジタル権利管理の解決法の多くは、電子著作物を暗号化して暗号化メッセージ及び復号鍵の双方を消費者のシステムに配布することに依存している。消費者から復号鍵を隠すために様々な方式が用いられているが、実際には、必要な情報は全て、悪意のあるユーザが電子著作物の保護を破るために利用可能である。現在の汎用コンピュータ及び消費者のオペレ

ーティングシステムが高度なセキュリティメカニズムとして提供されることが殆どないことを考慮すると、この脅威は現実であり、明白である。

【０００９】「セキュアコンテナー」（又は単に暗号化文書ともいう）は、許可条件のセットが要件を満たし、著作権料金（例えば、使用料の支払い）が認められるまで文書のコンテンツを暗号化したままにする方法を提供する。種々の条件及び料金を文書提供者と確認した後、文書を平文の形でユーザに公開する。IBM社のCryptolopeやInterTrust社のDigiboxなどの市販品はこの部類に属する。セキュアコンテナーのアプローチにより、安全でないチャネルを介する配送の際に文書を保護する解決法が提供されていることは明らかであるが、正当なユーザが平文の文書を入手し、コンテンツ所有者の著作権を違反してその文書を使用したり再配布したりするのを防ぐメカニズムが全く提供されていない。

【００１０】暗号化メカニズム及びセキュアコンテナーは、許可を受けたユーザ／購入者に電子著作物を転送する際に電子著作物を保護することに焦点を合わせている。しかしながら、電子著作物は、その使用においても、悪意のあるユーザや悪意のあるソフトウェアプログラムから保護されなくてはならない。たとえユーザが信頼されている人物であるとしても、そのユーザのシステムが攻撃を受ける可能性がある。電子著作物の電子商取引が直面する顕著な問題は、目標消費者のデバイス上での著作物の保護を保証する、ということである。電子著作物の保護が危うくなると、貴重で重要な情報が失われてしまう。今日の汎用コンピュータ及び消費者のオペレーティングシステムは安全性及び完全性の領域が不十分であるため、事が複雑になる。著作物をその使用にわたり保護することははるかに複雑な問題であり、この問題の大部分は未解決のままである。

【００１１】「高信頼性システム」のアプローチでは、システム全体が文書の無許可の使用及び配布を防ぐ責任がある。通常、高信頼性システムの構築は、安全なプロセッサ、安全な記憶装置、及び安全な解釈デバイスなどの新しいハードウェアの導入を伴う。この場合でも、高信頼性システムで実行する全てのソフトウェアアプリケーションの信頼性の高さが保証されなくてはならない。変更が困難な高信頼性システムの構築は未だに既存の技術に対するかなりの挑戦であり、現在の市場の傾向によると、PC及びワークステーションのようなオープンで信頼性の低いシステムが、著作権を所有する文書へのアクセスに用いられる主要のシステムになるであろうことが示唆されている。この意味で、一般的なオペレーティングシステム（例えば、Ｗｉｎｄｏｗｓ（登録商標）及びＵＮＩＸ（登録商標））及び解釈用アプリケーション（例えば、Microsoft社の Word）を備えたPC及びワークステーションのような既存のコンピューティング環境は高信頼性システムではなく、これらの構造を大幅に変えなくては信頼されることは不可能である。

【００１２】従って、信頼性の高い一定の構成要素を備えることはできるが、ユーザは、未知で信頼性の低い種々のエレメント及びシステムに頼り続けなくてはならない。このようなシステムでは、たとえ安全であることが期待されていても、予期せぬバグや弱点が見つかり、利用されることがよくある。

【００１３】従来の対称及び非対称の暗号化方法は、暗号化されるメッセージを、基本的にバイナリのストリングとして処理する。従来の暗号化方法を文書に適用すると、欠点がいくつか生じる。文書は、通常は比較的長いメッセージであり、長いメッセージの暗号化は、使用前に文書の復号を必要とするあらゆるアプリケーションの性能に大きな衝撃を与えうる。さらに重要なことに、文書は、表示、再生、印刷、そして編集さえも適切な解釈用アプリケーションに依存する、形式化されたメッセージである。一般に、文書を暗号化すると形式化された情報が破壊されるため、殆どの解釈用アプリケーションは、解釈前に文書を平文の形式に復号することを必要とする。解釈前に復号をすると、文書のインターセプトを望む者に復号ステップ後の平文の文書を公開してしまう可能性が生じる。

【００１４】権利の管理には、認証、許可、会計、支払いと金銭上の決済、権利の主張、権利の検証、権利の行使、及び文書の保護等様々な問題がある。この中でも、文書の保護は重要な問題である。ユーザがコンテンツ所有者の権利を認めて文書に対し特別な操作を行うことが許されている場合（印刷したり、画面に表示したり、音楽を演奏したり、ソフトウェアを実行する等）、文書は通常は平文である。つまり、暗号化されていない。簡単に説明すると、文書保護の問題は、文書がもっとも危険な状態（ユーザの管理下にあるマシン上で平文で記憶されている）のときにコンテンツ所有者の権利が侵されないようにすることである。

【００１５】文書が配布業者からユーザに安全に配布された（通常は暗号化された形式）場合でも、その文書を表示データ形式にしないと、ユーザは文書を表示したり操作したりすることはできない。したがって、十分な保護を実現するためには、最終的な段階でユーザに表示され、しかも使用可能な形式に戻しにくい形式で、文書のコンテンツを保護することが重要である。

【００１６】

【発明が解決しようとする課題】暗号化を使用する電子文書配布の既知の方法では、次のようないくつかのステップを経て処理される。まず、ユーザは暗号化文書を受け取る。次に、ユーザは自分の（公開鍵暗号化システムにおける）秘密鍵を使用してデータを復号し、文書の平文の内容を取り出す。最後に、平文の内容は解釈用アプリケーションに渡され、そのアプリケーションがコンピュータ可読文書を最終的な文書に変換し、ユーザのコン

ピュータ画面で表示したりハードコピーに印刷したりできるようにする。平文のコンテンツを解釈しなければならない理由は、解釈用アプリケーションは通常、サードパーティプロダクト（Microsoft社のWord（商標）やAdobe社のAcrobat Reader（商標）等）であり、入力される文書の形式がそのプロダクトに特有の形式であるからである。しかしながら、上記従来の文書の保護方法では、データを平文に復号化する第２のステップとコンテンツを解釈する第３のステップとの間で、それ以前は保護されていた文書であっても危険な状態に置かれる。つまり、復号されていて、しかも、ユーザのコンピュータ上に平文の電子形式で記憶されたままになっているからである。従って、ユーザが不注意だったりまたは経費を節約しようとする場合など、文書はコンテンツ所有者の許可を得ずに容易に再配布されてしまう可能性があるという問題点があった。

【００１７】どのシステムも不正を完全に防止したり攻撃の影響を受けたりしないわけではないが、近年の技術では、電子著作物の使用をユーザ指定の物理的デバイスに限定することによって電子著作物を保護するものがある。これらの技術により、ユーザは、電子著作物の解釈に使用する意図であるシステム又は物理的デバイスからのプライベート情報又はシステム状態情報を提供しなくてはならない。システム状態情報とは、通常、CPU識別子、デバイス識別子、NIC識別子、及びドライブ構造などのシステム構造情報として定義される。これらの技術において、電子コンテンツはセッション鍵を用いて暗号化され、次いで、ユーザの暗号鍵を用いずに、システム状態情報及びユーザの信用証明の組み合わせを用いて、セッション鍵を暗号化する。次に、暗号化されたコンテンツ及び鍵の双方を目的のリポジトリに送信する。受け取った暗号化著作物を使用するために、ユーザは信頼性の高い許可エンティティ（通常は遠隔に配置されたソフトウェアプログラム）に接触しなくてはならない。許可エンティティはユーザのアイデンティティ及び信用証明を確認し、次いでシステム状態を用いてセッション鍵を復号し、最後に使用のためにコンテンツを復号する。

【００１８】安全なAdobe社のAcrobat Readerや安全なMicrosoft社のMediaPlayerなどの市販のアプリケーションは、ライセンスバウチャーにおいて適切なユーザ信用証明及び使用権を調べることによって電子著作物の使用を有効にしている。ユーザの信用証明の中には、CPU識別子や一定のデバイスのシリアル番号などのシステムデバイス識別子がある。ユーザが電子著作物に対して操作を行う際、アプリケーションは指定のデバイスが存在するか否かを確認する。これにより、未許可のユーザ（実際は未許可のデバイス）には電子著作物が送信されないことが保証される。このプログラムのチェックによって最小限レベルの保証が提供されるが、このチェックはユーザのデバイスに存在する秘密の安全性に依存している。暗号鍵が侵されるのみでなく、デバイスの識別子自体も不正の脅威を特に受けやすい。

【００１９】Acrobat Reader及びMediaPlayerといった保護方式は、電子著作物のために発行されたライセンスバウチャーに指定されるユーザシステム上の必要なデバイスを解釈用アプリケーションに識別させることによって作動している。このことにより、多くの状況で（即ち、ユーザが信頼をおかれた者であり、ユーザの指定した解釈用デバイスが攻撃を受けやすくないものである場合）適切な保護のレベルが提供される。これらの方式の弱点は、暗号鍵の保護もライセンスバウチャーの完全性も侵されないであろう、という前提に基づいていることである。

【００２０】ユーザのアイデンティティ及び信用情報、並びにシステム状態情報が確認されるか又はライセンスバウチャーが受け取られると、コンテンツは平文に復号され、攻撃を受けやすくなる、という点で、これらの技術は実に保護技術というよりは認証技術である。電子著作物は、その使用にわたり保護を受けない。更に、ユーザ情報のアプローチは、ユーザが個人的な情報を流すのを十分に抑止することを仮定している、という点で問題がある。即ち、ユーザ情報のアプローチをうまく実行させるには、個人のアイデンティティ及び信用情報を公にするユーザに対して容赦のない結果をもたらさなければならない。

【００２１】特定のデバイスに許可を与える方式の大きな欠点は、重要な情報（例えば、CPU番号又は他の個人情報）を漏らすことをユーザに要求することであり、これにより、プライバシーの問題に関して懸念が生じる。ユーザは自発的に情報を漏らすが（ユーザがこの情報を漏らしたくない場合、ユーザの唯一のオプションは電子著作物を受け取らないことである）、個人情報を必要とせずにユーザのデバイス上の電子著作物を保護することのできる保護方式を提供することが望ましい。また、暗号鍵の保護やライセンスバウチャーの完全性に依存しないDRM解決法を提供することが望ましい。電子コンテンツの復号をできるだけ最後の段階まで遅らせるDRM解決法を提供することが望ましい。

【００２２】従って、既知のシステムの欠点を解消する電子文書配布方式を提供することが有益であろう。このような方式が提供されれば、ユーザは復号処理及び解釈処理時に、電子配布文書を再配布可能な形式で入手できなくなる。

【００２３】

【課題を解決するための手段】本発明の自己保護文書（ＳＰＤ）は、上記の従来技術の欠点に対処できる。自己保護文書は、暗号化文書と許可セット及びその暗号化文書を抽出して使用するために必要なソフトウェアの大部分が組み込まれた実行可能なコードセグメントとを組み合わせることで、特別なハードウェア及びソフトウェ

アを使用しなくても文書のコンテンツを保護できる。
【００２４】ＳＰＤシステムは、コンテンツ作成者（従来のモデルの著者及び出版社と類似したもの）とコンテンツ配布業者とに分けられる。著者／出版社はオリジナル文書を作成し、許可する権利を決定する。次に、配布業者が文書をカスタマイズして様々なユーザが使用できるようにするが、その過程で、ユーザの購入した許可範囲をユーザ自身が逸脱しないようにカスタマイズする。
【００２５】ユーザのシステムでは、自己保護文書は最後の段階で復号される。本発明の１実施形態では、ＳＰＤ自身に各種解釈機能も備わっている。そのため、このＳＰＤを使用すれば、信頼度が低い（また許可を受けずに使用することにもなる）外部アプリケーションに頼らなくても済む。別の実施形態では、サードパーティの解釈用アプリケーションのインターフェースとプロトコルを指定し、ＳＰＤと対話的に処理して解釈の信頼度を高めるようにしている。
【００２６】本発明の１実施形態では、暗号化文書はユーザシステムにより復号されるが、同時に、その文書はユーザシステムの状態に少なくとも一部依存している鍵により「秘話化（polarizing）」される。この秘話化は暗号の面からは配布に使用される暗号化処理に比べ安全度は低いが、偶発的なコピーの抑止には役立つ。本発明では、解釈の処理時及びそれ以降に秘話解除が行われ、その結果、文書の中間形式は実質上使用不能になる。
【００２７】本発明の別の実施形態では、電子著作物の保護方法は盲目的変換関数を使用し、暗号化された電子著作物を暗号化プレゼンテーションデータに変換する。発信者の電子コンテンツは、復号されないことにより元の形式で保護される。この方法により、解釈用又は再生用アプリケーションは、初めに暗号化文書を復号せずに、暗号化文書を処理して暗号化プレゼンテーションデータにすることができる。次に暗号化プレゼンテーションデータは復号され、その直後にユーザに表示される。この方法は、復号オーバーヘッドを最小にし（解釈前の復号は一般に時間及びリソースをより多く消費するため）、復号を解釈処理の最後の段階まで延ばすことにより、（復号及び解釈双方の）処理の全体的な性能を向上させる。
【００２８】盲目的変換又は盲目的コンピューティングは、複数の方法のうち一つの方法で達成することができる。電子著作物の殆どが形式情報を含み、これは暗号化の際に再生用又は解釈用アプリケーション（電子著作物をプレゼンテーションデータに変換する変換関数）によって処理することができない。電子著作物が、形式を保つ暗号化方式で暗号化される場合、あらゆる変換関数を用いることができる。このことは、どの市販の再生用又は解釈用アプリケーションも暗号化された電子著作物を処理して暗号化プレゼンテーションデータにすることができる、という点で特に有用である。その他に、盲目的変換関数は、元の変換関数の一関数である。例えば、盲目的変換関数を、元の変換関数の多項式関数とすることができる。あるいは、盲目的変換関数及び元の変換関数の双方を、任意の多変量の整数係数であるアフィン関数とすることができる。
【００２９】全ての暗号化方式が、形式を保つ暗号化方式であるわけではない。加法的暗号化方式を、全ての文書タイプ及び全ての関連変換関数に用いることができる。いくつかの再生用又は解釈用アプリケーションでは、あるタイプの文書に対しては、形式情報の部分が平文のまま残される場合がある。他のタイプの文書では、全ての形式情報を暗号化することができる。あるタイプの文書では、加法的暗号化方式を用いて形式情報を暗号化することができ、あらゆる暗号化方式を用いて文書のコンテンツ又はデータ部分を暗号化することができる。
【００３０】特に、加法的暗号化方式を用いて文書の座標情報を暗号化することができ、これにより、暗号化された座標データに対してある程度の解釈変換を行うことができるようになる。例えば、特殊な種類の文書やトークンベースの文書では、形式保存暗号化の際に暗号化方式を用いる場所は２つある。即ち、一方は文書内の特定のトークンの座標、即ち位置情報x及びy用であり、もう一方は個々のトークン画像の辞書用である。文書内の特定のトークンの個々の座標に対して盲目的変換を行うには、初めの暗号化方式は加法的暗号化方式でなくてはならない。しかしながら、トークン辞書はあらゆる暗号化方式で暗号化することができる。
【００３１】暗号化されたトークン辞書からは、なおトークン画像のサイズなどの情報が漏洩する場合がある。このことが問題になる場合（トークン辞書が小さい場合など）は、暗号化の前にトークンをいくつかの余分なビットで埋め込む(pad)ことができる。この埋め込みにより、同一サイズ又はいくつかの固定サイズの暗号化トークン画像を生じることができる。トークンベースの文書に対しては、辞書内のトークンの座標情報を符号化しない場合がある。例えばハフマン符号語としての座標の符号化を望む場合、識別子の暗号化に用いられるのと同一のアプローチを用いてこの状況に対処することができる。基本的には、位置テーブル内の符号語を平文のままにし、符号語辞書内の符号語を、ある一方向ハッシュ関数を用いてハッシュし、対応する座標情報を暗号化する。解釈の際は、まず位置テーブル内の符号語をハッシュし、次に、暗号化された座標情報の探索に用いる。
【００３２】本発明の別の実施形態では、電子著作物及びシステムコンテクスト（又はリソース情報又はシステムリソース）を秘話化し、電子コンテンツを秘話解除することなく電子著作物の高信頼性の解釈又は再生を可能にする。この実施形態では、電子著作物は電子コンテンツ及びリソース情報を含むタイプのものである。リソース情報は、電子著作物をプレゼンテーションデータにフ

ォーマット又は処理するために再生用アプリケーションが用いる情報を含みうる。リソース情報は、例えば、フォントテーブル、色パレット、システム座標及び音量設定など、特定のシステム上の再生用ソフトウェアに利用できるシステムリソースの集まりを含みうる。

【００３３】様々なタイプの電子著作物を秘話化することができる。一般的な文書タイプの電子著作物の秘話化に加えて、オーディオ及びビデオの電子著作物を秘話化することができる。通常、電子著作物及びシステムコンテクストは、秘話化エンジンを用いて製造者又はコンテンツ所有者の場所で秘話化される。秘話化エンジンは、電子著作物及びシステムコンテクストを各々の秘話化形式に変換するのに用いられる構成要素である。秘話化エンジンは、秘話化エンジンの初期化及びカスタマイズに用いられるエレメントである秘話化シードに依存する秘話化方式を用いる。

【００３４】種々の秘話化方式を用いて電子著作物を秘話化することができる。例えば、無状態の秘話化は乱数をシードとして用い、電子著作物を秘話化された電子著作物に変換する。状態ベースの秘話化方式は、システムの状態又は特性に基づいたシードを用い、電子著作物を、そのシステム状態又は特性に関連づけられた秘話化電子著作物に変換する。動的状態ベースの秘話化方式は、動的なシステムの状態又は特性に基づいたシードを用い、電子著作物を秘話化された電子著作物に変換する。この実施形態では、秘話化された電子著作物には一般に秘話化エンジンが備えられており、この秘話化エンジンは、システムが電子著作物の再生を要求する度に、符号化された電子著作物及び符号化されたシステムコンテクストを動的状態ベースの秘話化方式に従って再び秘話化する。許可ベースの秘話化方式は、信頼のおけるソースから受け取った許可情報に基づいたシードを用い、電子著作物を秘話化された電子著作物に変換する。安全性を更に高めるために、秘話化されたシステムコンテクストを、秘話化された電子著作物とは別に、取り外し可能なコンテクストデバイスに保存することができる。このデバイスは、電子著作物を使用する前にシステムに接続される必要がある。

【００３５】秘話化シードは、最終的なエンドユーザ又は最終的なエンドユーザシステムに特定の電子著作物を関連づけるために用いることのできる情報を含むことが好ましい。一般に、所有者又は配布業者は、電子著作物の秘話化に使用する秘話化方式のタイプと、使用する秘話化鍵のタイプとを、電子著作物の価値によって選択する。暗号化方式と同様、秘話化方式は複雑性及び強度のレベルが様々である。電子著作物が注文されると、システムコンテクストと呼ばれる、その電子著作物のリソース情報の一部のコピーを作成する。秘話化シードを選択し、電子著作物及びシステムコンテクストの双方を秘話化する。電子著作物に対して使用されるものとは異なる秘話化方式を、システムコンテクストに対して使用してもよい。しかしながら、秘話化シードは双方とも同一のものである。次に、秘話化された電子著作物及び秘話化されたシステムコンテクストをユーザに提供し、ユーザは再生又は解釈システムにおいて再生又は解釈を行う。

【００３６】形式保存暗号化及び高信頼性解釈を提供する本発明の実施形態では、暗号化されたプレゼンテーションデータを平文のプレゼンテーションデータに復号する必要性が生じるまで保護が提供される。本発明のこの実施形態では、再生用アプリケーションは秘話化されたリソース情報を用いて、秘話化電子著作物を平文のプレゼンテーションデータに変換する。

【００３７】電子著作物の電子コンテンツのみを秘話化してリソース情報を秘話化しないでおく、即ち平文のままにしておく場合でも、再生用アプリケーションは、秘話化された電子著作物を処理して秘話化されたプレゼンテーションデータにすることができる。このことは、秘話解除部がプレゼンテーションデータの秘話解除を行い、ユーザによる表示又は使用に好適な平文のプレゼンテーションデータにしなくてはならないことを意味している。電子著作物のリソース情報の一部もこれに従って秘話化される場合は、再生用アプリケーションが秘話化された電子著作物を変換する際に、再生用アプリケーションは秘話化されたシステムリソース情報を用いて、秘話化された電子著作物を平文のプレゼンテーションデータに変換する。必要とされるリソース情報を全て秘話化してもよいし、その一部のみを秘話化してもよい。再生用アプリケーションが、元の秘話化されていない電子コンテンツを見ていない、という点で再生は盲目的である。

【００３８】この実施形態では、秘話化されたシステムコンテクスト（リソース情報）を用いて、秘話化された電子著作物を再生用アプリケーションによって変換し、平文のプレゼンテーションデータを生成する。再生用アプリケーションは、あらゆる市販の又はサードパーティのアプリケーションが可能である。再生用アプリケーションは、プレゼンテーションデータを秘話解除するためにカスタマイズされる必要はなく、秘話解除部のエンジンも不要である。再生用アプリケーションは、（秘話化されたシステムリソースを用いて秘話化された電子コンテンツを処理する）盲目的再生システムとして作動し、電子著作物を変換又は符号化する秘話化のタイプに依存する。これにより、ソフトウェアプログラムを用いて電子著作物を再生する能力が特定のリソース情報に関連づけられ、従って使用にわたりコンテンツが保護される。

【００３９】暗号化を用いて電子著作物を保護し、最後に電子著作物を復号して平文の形式にし、それから電子著作物を再生用アプリケーションに提供するシステムとは異なり、盲目的再生システムは、再生処理のできる限り後段まで電子著作物を秘話化の形式で符号化したまま

にする（盲目的な再生には、はっきりとした復号ステップはない）。盲目的再生システムでは、秘話化された電子著作物そのものは、平文に秘話解除されない。プレゼンテーションデータの品質は一般にオリジナルの電子著作物よりも劣るため、たとえプレゼンテーションデータが平文の形式で取得されても、オリジナルの電子著作物に（変換されるとしても）容易に変換することはできない。

【００４０】多くの様々なタイプの電子著作物及びそのリソース情報を秘話化して、盲目的再生システムで再生することができる。文書、テキスト、オーディオファイル、グラフィックファイル及びビデオファイルなどの電子著作物を、適切なリソース情報の秘話化により、本発明の盲目的再生システムで再生することができる。

【００４１】本発明の構造及び機能は、本明細書と共に含まれる図面を参照することで最も良く理解される。

【００４２】

【発明の実施の形態】本発明の実施の形態について図面を参照しながら説明する。本発明を様々な形で具体化することができ、その中には、開示される実施の形態の形とは非常に異なりうるものもあることは明白であろう。結果的には、本明細書において開示される特定の構造及び機能の詳細は代表的なものであるにすぎず、本発明の範囲を限定するものではない。

【００４３】図１は、文書の電子配布のシステムの最上位機能モデルを表している。上記で定義したように、これらの文書には、通信文書、書籍、雑誌、ジャーナル、新聞、他の書類、ソフトウェア、オーディオ及びビデオクリップ、及び他のマルチメディアプレゼンテーションが含まれる。

【００４４】著者（または出版社）１１０は文書のオリジナルコンテンツ１１２を作成し、配布業者１１４に渡して配布する。著者が他人を配布業者として使用せずに文書を直接配布することも考えられるが、図１に示したように作業を分割すると効率は向上する。それは、著者／出版社１１０が、配布業者１１４が行う機械的で平凡な役割ではなく、コンテンツの作成に集中できるからである。さらに、このように作業を分担することで、配布業者１１４も多数の著者及び出版社（図示されている著者／出版社１１０も含め）と連携することで、規模の節約を実現できるからである。

【００４５】次に、配布業者１１４は、変換されたコンテンツ１１６をユーザ１１８へ渡す。標準的な電子配布モデルでは、変換されたコンテンツ１１６はオリジナルコンテンツ１１２の暗号化版を表している。つまり、配布業者１１４はユーザ１１８の公開鍵を使用してオリジナルコンテンツ１１２を暗号化し、変換されたコンテンツ１１６は特定のユーザ１１８のためにだけカスタマイズされる。次に、ユーザ１１８は自分自身の秘密鍵を使用して変換されたコンテンツ１１６を復号すれば、オリジナルコンテンツ１１２を表示できる。

【００４６】コンテンツ１１２に対する支払い１２０は、ユーザ１１８から配布業者１１４へ決済機関１２２を介して渡される。決済機関１２２は、ユーザ１１８から、及び特定の文書の表示を希望する他のユーザから要求を収集する。決済機関１２２は、支払い取引、クレジットカード取引、及び他の既知の電子支払い方式等の支払い情報も収集し、収集したユーザの支払いを配布業者１１４へ支払いバッチ１２４として送信する。もちろん、決済機関１２２はユーザの支払い１２０の分け前の一部を受け取る。また配布業者１１４も、支払いバッチ１２４の一部を受け取った上で、著者と出版社１１０へ支払い１２６（印税も含む）を送信する。この方式の１実施形態では、配布業者１１４は特定の１文書に関するユーザ要求をまとめてから送信する。このようにすると、変換されたコンテンツ１１６が含まれた１文書を、すべての要求ユーザが復号できるように生成できる。この生成方法は、本分野では既知である。

【００４７】また、ユーザ１１８が文書を要求（または使用）するたびに、会計メッセージ１２８を監査サーバ１３０へ送る。監査サーバ１３０は、ユーザ１１８の各要求が配布業者１１４が送信する文書と一致することを確認する。そのために、監査サーバ１３０は、会計情報１３１を配布業者１１４から直接受け取る。矛盾が生じたら、その矛盾を報告書１３２を介して決済機関１２２へ送る。これにより決済機関では配布業者１１４へ送る支払いバッチ１２４を調整できる。このような会計方式が確立されているため、この電子文書配布モデルでは詐欺行為が行われる確率が低く、また、使用時間または使用度に応じて料金が変わる時間依存型の使用許可を扱うこともできる。

【００４８】図１に示した上記の文書の電子商取引のモデルは、現在一般的に使用されているものである。以下で詳細に説明するように、このモデルは自己保護文書の配布用に説明するシステム及び方法にも同様に適用される。

【００４９】図２には、電子文書配布に関する従来技術のシステムでユーザ１１８（図１）が実行するステップが示されている。上記で説明したように、通常は暗号化装置を使用して文書を暗号化する。次に、暗号化されたこれらの文書をパブリックに配布及び保管し、許可されたユーザがプライベートに復号する。この形式は、文書の配布業者から目的ユーザまでパブリックネットワークを介して文書を配送したり、安全でない媒体上に文書を記憶したりするときの保護の基本形式である。

【００５０】最初に、ユーザ１１８が暗号化文書２１０を受け取り、復号ステップ２１２へ移る。この分野では既知のように、復号ステップ２１２ではユーザ１１８の秘密鍵を受け取る。この鍵は、ユーザのコンピュータにローカルに記憶しておくか、または必要に応じユーザが

入力する。文書２１０を復号すると、オリジナルコンテンツ１１２（図１）と類似または一致する平文コンテンツ２１６が生成される。

【００５１】平文コンテンツ２１６が解釈用アプリケーション２１８へ渡されると、このアプリケーションはプレゼンテーションデータ２２０（つまり、文書のオリジナルコンテンツ１１２の使用可能版）を作成する。通常、このようなシステムでは、プレゼンテーションデータ２２０は文書タイプに従って、ただちにビデオ画面に表示したり、ハードコピーとして印刷したり、または他の目的で使用したりできる。

【００５２】上記で説明したように、このようなシステムでは文書に弱点がある。平文コンテンツ２１６は、配布業者１１４または著者／出版社１１０の承諾または同意なしに、コピー、記憶、または他のユーザへの譲渡が可能である。また、正当なユーザでも、コンテンツ所有者の所有権に配慮せず、文書を平文の形式で受け取って自由に再配布及び使用することでライセンス料の節約を図ろうとするユーザがいる。既に説明したように、本発明では、ユーザシステムでの解釈の処理時に、ユーザが文書を再配布可能な形式で入手できない方式を提供している。

【００５３】したがって、本発明のシステム及び方法では、ユーザ１１８のシステムにおいて暗号化文書を処理する別の方式を提供している。この方式の簡単な実施例を図３に示す。

【００５４】図３は、暗号化文書３１０が復号ステップ３１２（秘密鍵３１４を使用する）及び解釈用アプリケーション３１６へ渡され最終的にプレゼンテーションデータ３１８が作成されるという点において、図２と似ている。しかし、保護シェル３２０により、保護層が別個に用意されている。保護シェル３２０が提供されているため、平文コンテンツを取り込み可能（インターセプト可能）な状態にせずに（図２の平文コンテンツ２１６のように）、文書３１０を復号及び解釈できる。これは、以下に図５を参照して説明するように、文書３１０に復号要素及び解釈要素を組み込むことで実現する。組み込む復号及び解釈要素はユーザのＳＰＤとの対話を制限するように調整され、ユーザ許可に応じて特定の操作（文書の保存またはカットアンドペースト操作の実行）等を制限する。

【００５５】図４はさらに高度なバージョンである。図４の方式には中間「秘話化」（polarizing）ステップ、すなわち、簡易暗号化のステップが含まれていて、復号後で解釈前の文書の安全を確保するように改良されている。まず、暗号化された文書コンテンツ４１０は秘話化部４１２へ渡される。秘話化部４１２はユーザの秘密鍵４１４を受け取り、復号ステップ４１６を介して、文書コンテンツ４１０を復号する。同時に、秘話化部４１２はユーザのシステムから秘話化鍵４１８を受け取る。

【００５６】秘話化部４１２は、この秘話化鍵４１８を使用し、文書を秘話コンテンツ４２０を内容とするバージョンへ変換する。これらの操作はすべて、秘話化部４１２が文書の復号と秘話化処理との間で文書の平文バージョンを記憶していない限り、保護機構を使用せずにオープンで行うことができる。

【００５７】本発明の１実施形態では、秘話化鍵４１８はユーザシステムの内部状態から取り出したデータ要素を組み合わせたものを表している。これらのデータ要素には、日付と時刻、最後のキーストロークからの経過時間、プロセッサの速度とシリアル番号、及びユーザシステムから継続的に取り出すことができる他の情報等が含まれる。秘話化鍵４１８には、秘話化コンテンツ４２０を取り込んだり獲得したりしてもそのコンテンツが役立たなくなるような時間関連情報を組み込んでおくと便利である。このようにしておけば、システム時間は大幅に変わるため、秘話化文書の解釈は不可能になる。

【００５８】次に、再び保護シェル４２２内で、秘話化されたコンテンツ４２０は解釈用アプリケーション４２４へ渡される。上記で説明したように、標準的な解釈用アプリケーションとしては、Microsoft社のWord（商標）またはAdobe社のAcrobat Reader（商標）等のサードパーティ・アプリケーションがある。しかし、このような外部の解釈用アプリケーションは、秘話化されたコンテンツ４２０を処理できない場合も考えられる。これは、コンテンツ、フォーマットコード、及び解釈処理側で使用する指示符号が秘話化処理時にスクランブルされるからである。

【００５９】したがって、解釈用アプリケーション４２４には互換性（又は少なくともフォルト・トレラント性）が要求され、若しくは、ほぼ完全にアプリケーションが処理可能な秘話化コンテンツ４２０を受け取らなければならない。後者の可能性については、図９と関連して以下に説明する。

【００６０】解釈用アプリケーションの出力は秘話化プレゼンテーションデータ４２６（秘話化解釈コンテンツ）で、これは解釈用アプリケーション４２４によりフォーマットされているが、まだ秘話化されたままであるため、ユーザがそのまま読み取ることはできない。秘話化プレゼンテーションデータ４２６は秘話解除部４２８に渡され、その秘話解除部が秘話化鍵４１８を受け取って文書の元の形式をプレゼンテーションデータ４３０（平文解釈コンテンツ）として復元する。本発明の１実施形態では、この秘話解除機能は解釈の機能または表示機能と組み合わされている。この場合、秘話化プレゼンテーションデータ４２６は表示装置が直接受け取る。この表示装置はユーザシステムと別個のもので、通信チャネルを介してデータを受け取るものであっても構わない。

【００６１】秘話化鍵４１８の作成、解釈用アプリケー

ション４２４、及び秘話解除ステップ４２８は、すべて保護シェル４２２の構成要素である。これらは変更が困難なプログラム要素である。保護シェル４２２の内部で実行されるすべての計算（又は変換）ステップはローカルデータだけを使用し、グローバルにアクセス可能な記憶媒体やメモリー領域へは一時データを格納しない。最終的に明示できる結果だけを保護シェル４２２からエクスポートする。この方法により、中間データをインターセプトしたり、利用したりする目的で、ユーザが簡便な手法をとることができなくなる。例えばオペレーティングシステムのエントリー・ポイントを修正したり、システム資源を窃取したりすることができなくなる。

【００６２】本発明の別の実施の形態では、図４のプレゼンテーションデータ４３０はデバイス非依存型データまたはデバイス依存型データのいずれでも構わない。デバイス非依存型の場合、解釈処理を完了するためには、通常、デバイスドライバ（ディスプレイドライバまたはプリンタドライバ等）による追加処理が必要になる。現時点での好ましいデバイス非依存型データの場合、プレゼンテーションデータに対する各デバイスへの適合補正は（解釈用アプリケーション４２４または秘話解除ステップ４２８のいずれかで）すでに行われていて、プレゼンテーションデータ４３０を目的の出力装置に直接出力できる。

【００６３】図３及び図４を使用して説明した上記の復号方式は、図５で詳細に示している独自の文書のデータ構造により実現される。上記で説明したように、本発明のシステム及び方法が実行する特定の操作では、高信頼性の構成要素が必要である。特定の純正コード（修正されていないコード）を使用して本発明の信頼性を向上させる方法の１つは、このコードを文書と共に提供することである。かかる方法を具現化する本発明による自己保護文書の各種データ構成要素については、図５で説明する。

【００６４】本発明による文書保護の問題解決方法は、ユーザシステム側で高信頼性ハードウェア装置またはソフトウェアモジュールを用意していないという前提で使用される。これを実現するために、文書の機能を強化し、アクティブなメタ文書オブジェクトにする。コンテンツ所有者（つまり著者または出版社）は、文書に権利情報を付加し、使用目的のタイプ、必要な許可と関連料金、及びユーザに許可を与えるソフトウェアモジュールを指定する。文書と、関連する権利と、権利の行使を実現する付加ソフトウェアモジュールを組み合わせたものが、本発明にいう自己保護文書（ＳＰＤ）である。自己保護文書では許可されていなかったり、想定されていない管理外の使い道や文書の配布が防止されるため、コンテンツ所有者の権利が保護される。

【００６５】自己保護文書５１０は、次の３種類の主要機能セグメントにより構成されている。実行可能コードセグメント５１２には、ユーザが暗号化文書を使用するために必要な実行可能コード部分が含まれている。権利及び許可セグメント５１４には、さまざまなユーザに許可する各種アクセスレベルを表すデータ構造体が含まれている。コンテンツセグメント５１６には、ユーザが表示する暗号化コンテンツ１１６（図１）が含まれている。

【００６６】本発明の好適な実施形態では、ＳＰＤ５１０のコンテンツセグメント５１６は、文書メタ情報５１８（文書のタイトル、フォーマット、及び改訂日等の情報）、権利ラベル情報５２０（テキストと共に表示する著作権の表示と権利及び許可情報）、及び保護コンテンツ５２２（暗号化された文書自身）の３種類のサブセクションで構成される。

【００６７】本発明の１実施形態では、権利及び許可セグメント５１４には、各許可ユーザごとの権利情報が含まれる。料金及び条件の一覧を、各ユーザの権利に加えても構わない。例えば、John Doeというユーザに特定の文書を表示する権利と、２回だけ印刷する権利とを１０ドルで与えることができる。この場合、権利及び許可セグメント５１４ではJohn Doeを識別し、彼に２種類の権利を関連付け（表示権及び印刷権）、価格（１０ドル）及び印刷の制限（２回）等の料金と条件を指定する。権利及び許可セグメント５１４には、他のユーザの情報を組み込んでも構わない。

【００６８】別の実施形態では、権利及び許可セグメント５１４には権利情報を指定する外部情報へのリンクだけを組み込む。この場合、実際の権利及び許可はネットワークで接続された許可サーバ等の別の場所に記憶されていて、文書を使用するたびに照会を行う必要がある。この方法では、権利及び許可をコンテンツ所有者が動的に更新できるという利点がある。例えば、表示のための価格を引き上げたり、許可されていない態様での使用を検出したらユーザの権利を無効にしたりできる。

【００６９】いずれの場合にも、権利及び許可セグメント５１４は暗号で署名し（本技術分野では既知の方法により実現できる）、指定された権利及び許可を不正に変更できないようにするのが好ましい。また、ユーザが自分自身及び他人の権利及び許可を直接表示できないように暗号化することも好ましい。

【００７０】実行可能コードセグメント５１２（「ＳＰＤ制御」とも呼ばれる）にも幾つかのサブセクションが含まれていて、各サブセクションは、少なくとも一部が実行可能コードセグメントに含まれるソフトウェアモジュールで構成されている。本発明の１実施形態では、このＳＰＤ制御にJavaプログラミング言語を使用している。しかし、本発明を実現するには、プラットフォームに依存しない言語であるかプラットフォームに固有の言語（インタプリタ型またはコンパイラ）であるかに関わらず、任意の言語を使用できる。

【００７１】権利行使部５２４は、ユーザのＩＤを確認し、ユーザが要求するアクションと権利及び許可セグメント５１４に列挙されているアクションとを比較し、指定された権利に基づいて要求されたアクションを許可または拒否するために用意されている。権利行使部５２４の処理は、図７を参照して以下に詳細に説明する。

【００７２】秘話化エンジン５２６も、実行可能コードセグメント５１２に、保護された状態で含まれている。このエンジンは、既に説明したように、システムの状態（または他の秘話化鍵）に従ってデータを読み取り、秘話化する。本発明の好適な実施形態では、秘話化エンジン５２６は文書の記憶前または復号前にその文書を処理するため、ユーザシステムに文書が平文で記憶されることはない。秘話化エンジン５２６は保護されていて（つまり、暗号署名及び暗号化されていて）、変更、リバースエンジニアリング、及び逆アセンブルできないようになっている。

【００７３】対応する秘話解除エンジン５２８も実行可能コードセグメント５１２に含まれていて、秘話化コンテンツから平文のプレゼンテーションデータを生成できるようにしている（図４を参照されたい）。秘話解除エンジンにはセキュアウィンドウオブジェクトの組が含まれていて、ユーザシステムの解釈用ＡＰＩ（Application Program Interface）に対する変更防止インターフェースになっている。セキュアウィンドウオブジェクトはインターセプトが困難である。このため、オペレーティングシステム用のデータをインターセプト、又は受信して平文形式の文書を再構築する機会を少なくできる。

【００７４】実行可能コードセグメント５１２に含まれている、対応する秘話解除エンジン５２８は、秘話化されたコンテンツから平文のプレゼンテーションデータを生成できる（図４を参照されたい）。また、この秘話解除エンジン５２８は、論理出力装置または物理出力装置（例えば、ユーザの表示装置）に対する変更防止インターフェースである。秘話解除エンジン５２８に入力されるのは、秘話化プレゼンテーションデータである。したがって、そのデータがインターセプトされても、ユーザのシステム状態等に依存する秘話解除の処理を実行しないと平文コンテンツは得られない。

【００７５】セキュア表示部５３０は、実行可能コードセグメント５１２にオプションで組み込まれる。セキュア表示部５３０は、権利及び許可セグメント５１４に基づき、許可されているアクセスレベルだけを許可するために使用される。例えば、ユーザが文書を表示する権利しか買っていない（保存（セーブ）や印刷の権利は買っていない）場合には、表示部は、ユーザに対し保存や印刷は許可せず、また、現在の大部分のオペレーティングシステムで実行可能なカットアンドペーストの実行も許可しない。

【００７６】また、解釈用エンジン５３２が実行コードセグメント５１２に含まれているか、または、実行コードセグメント５１２により参照される。解釈用エンジン５３２は、保護する必要はない。したがって、解釈用エンジン５３２のコードはＳＰＤアプレット内に組み込まれていてもよいし、他の場所から（セキュアリンクを介して）取得することとしても構わない。どちらの場合も、解釈用エンジン５３２は、秘話化文書コンテンツの入力を受けて、当該コンテンツデータから秘話化プレゼンテーションデータを作成するように設定されている（図４を参照されたい）。

【００７７】自己保護文書５１０の上記の態様及び要素を、システムの動作と共に、以下に詳細に説明する。

【００７８】図６は、自己保護文書５１０が作成され、配布されるときに実行されるステップを示したものである。汎用（generic）ＳＰＤ６１０には、ユーザ固有の権利情報は組み込まれておらず、特定のユーザ用に暗号化もされていない。汎用ＳＰＤ６１０は、３つの項目、すなわち、平文（暗号化されていない）形式のオリジナル文書コンテンツ６１２、高レベル権利指定６１４、及びオプションの電子透かし６１６から作成される。

【００７９】コンテンツ６１２は、著者または出版社の希望に合わせて、文書のレイアウトを決定するように事前処理（プリプロセス）される（ステップ６１８）。例えば、希望するページサイズ、フォント、及びページレイアウトを選択できる。コンテンツ６１２は、ユーザシステム及びＳＰＤと互換性がある形式になるように、コンテンツ事前処理ステップで「事前解釈」される。例えば、コンテンツ６１２はMicrosoft Word（「.DOC」）またはAdobe Acrobat（「.PDF」）形式から解釈用エンジン５３２が読み取れるように特別に設定された別の形式に変換される（図５）。本発明の１実施形態では、コンテンツ６１２の複数のバージョンがコンテンツ事前処理ステップで生成され、汎用ＳＰＤ６１０に記憶される。ユーザは、これらの異なったバージョンを、要求に応じて個別に購入できる。

【００８０】高レベル権利指定６１４では、アクセス権利の可能な組合せを記述する。この権利指定は、文書ごとに合わせて設定され、下流のユーザの様々なクラスの様々な権利グループを記述できる。例えば、１コピー当り１．００ドル、追加コピーに２．００ドルの使用料で最大１００，０００部の文書を配布する権利を出版社に与えることができる。同様に、１ヶ月後や１年後に「期限切れ」する文書または期限のない文書等、各バージョンの文書購入オプションをユーザに与えることができる。考えられるいくつかの制限について、詳細な例を参照して説明する。以下に例を述べる。

【００８１】Digital Property Rights Language（ＤＰＲＬ）は、デジタル著作の権利を指定するために使用される言語である。この言語は、権利に関する各種料金及び条件を指定し、権利を行使する機能を提供している。

権利指定は、ＤＰＲＬのステートメントとして表現される。詳細については、Stefikに付与された米国特許第５，７１５，４０３号、「System for Controlling the Distribution and Useof Digital Works Having Attached Usage Rights Where the Usage Rights areDefined by a Usage Rights Grammar」等を参照。権利の行使及び権利に関連する条件の検証は、ＳＰＤ技術を使用して行われる。

【００８２】各種権利は、「ｗｏｒｋ（ワーク）」指定を使用してデジタル著作物の各要素について指定できる。ｗｏｒｋ指定では、各著作に適用可能な各種の権利のセットを指定できる。権利は、「right group」と呼ばれる名前付きグループに分類できる。権利グループ内の各権利は、条件セットに関連付けられる。条件には、支払う料金、使用時間、アクセスタイプ、電子透かしタイプ、処理を行う装置タイプ等様々な種別がある。ＤＰＲＬでは、譲渡、表現権、派生著作権、ファイル管理権、及び構成権等の各種権利カテゴリに対応している。トランスポート権は、ある格納場所（リポジトリ）から別のリポジトリへの著作物の移動に関する。表現権は、著作物の印刷及び表示、より一般的には、変換装置を介して著作物を外部媒体へ送信することに関する（これには、平文のコピーを作成するために使用する「エクスポート」権も含まれる）。派生著作権は、新しい著作物を作成する場合に著作物の再使用をすることに関する。ファイル管理権は、バックアップコピーの作成及び復元に関する。また、構成権はリポジトリのソフトウェアのインストールに関する。

【００８３】ＤＰＲＬのワーク指定の例を以下に示す。

【００８４】

【外１】

```
(Work:
(Rights-Language-Version: 1.02)
(Work-ID: "ISDN-1-55860-166-X; AAP-2348957tut")
(Description: "Title: 'Zuke-Zack, the Moby Dog Story'
        Author: 'John Beagle'
        Copyright 1994 Jones Publishing")
(Owner: (Certificate:
                (Authority: "Library of Congress")
                (ID: "Murphy Publishers")))
(Parts: "Photo-Celebshots-Dogs-23487gfj" "Dog-Breeds-Chart-AKC")
(Comment: "Rights edited by Pete Jones, June 1996.")
(Contents: (From: 1) (To: 16635))
(Rights-Group: "Regular"
(Comment: "This set of rights is used for standard retail editions.")
(Bundle:
(Time: (Until: 1998/01/01 0:01))
        (Fee: (To: "Jones-PBLSH-18546789")(House: "Visa")))
(Play:
        (Fee: (Metered: (Rate: 1.00 USD) (Per: 1:0:0) (By: 0:0:1))))
(Print:
        (Fee: (Per-Use: 10.00 USD))
        (Printer:
                (Certificate:
                        (Authority: "DPT"
                        (Type: "TrustedPrinter-6")))
         (Watermark:
                (Watermark-Str: "Title: 'Zeke Zack - the Moby Dog' Copyright
                1994 by Zeke Jones, All Rights Reserved.")
                (Watermark-Tokens: user-id institution-location render-name
                render-time))))
(Transfer: )
(Copy: (Fee:   (Per-Use: 10.00 USD)))
(Copy: (Access:
                (User: (Certificate:
                                (Authority: "Murphy Publishers")
                                (Type: "Distributor")))))
(Delete:)
(Backup:)
(Restore:       (Fee:   (Per-Use: 5.00 USD)))))
```

【００８５】このｗｏｒｋ指定には「Regular」と呼ばれる権利グループがある。「Regular」は、「Zuke-Zack, the Moby Dog Story」という題名の書籍の標準小売版の権利を指定している。このワーク指定は、表示再生（ｐｌａｙ）、印刷（ｐｒｉｎｔ）、転送（ｔｒａｎｓｆｅｒ）、コピー（ｃｏｐｙ）、削除（ｄｅｌｅｔｅ）、バックアップ（ｂａｃｋｕｐ）、及びリストア（ｒｅｓｔｏｒｅ）等の幾つかの権利の条件を表している。この例の著作物には、他のソースから組み込まれた、さらに２つの構成要素として、写真と犬の種類表（chart of breeds）とが含まれている。「bundle」指定は、グループ内のすべての権利に適用される共通の条件セットをまとめている。この指定は、グループ内のすべての権利が１９９８年の１月１日まで有効で、料金をアカウント「Jones-PBLSH-18546789」に支払うことを表している。この取引の決済機関はＶｉｓａである。さらに以下に述べる契約が適用される。著作物の再生には１時間当り１．００ドル支払い、料金は秒単位で累算される。著作物は「ＤＰＴ」により保証されるTrustedPrinter-6で印刷でき、１回の印刷当り１０．００ドルの料金である。印刷されたコピーには、（上記のように設定された）電子透かし文字列と印刷時に分かっている「指紋（finger print）」としてのトークンリストとを付ける。この著作物は、１０．００ドル支払うかまたはＭｕｒｐｈｙ出版から配布業者証明書を入手してコピーできる。この著作物の無制限の譲渡、削除、またはバックアップが許されている（リストア・コスト５．００ドル）。

【００８６】高レベル権利指定６１４も事前処理ステップの対象になる（ステップ６２０）。この場合、高レベル（人間が読み取り可能な）指定は、より効果的なデータ構造表現にコンパイルされ、本発明で使用できるような形式になる。

【００８７】次に、事前処理されたコンテンツ６１２、事前処理された権利指定６１４、及び電子透かし６１６を組み合わせることで汎用ＳＰＤ６１０を作成する（ステップ６２２）。電子透かしは、本技術分野で知られている任意の方法で付加できる。ＳＰＤにおける電子透かしは、目で確認できる形式でも、できない形式でもよい。汎用ＳＰＤ６１０は、著者／出版社１１０によりオプションで暗号化し、配布業者１１４へ送信してもよい（図１）。

【００８８】次に、配布業者１１４は汎用ＳＰＤ６１０を受け取り、後でカスタマイズできるように記憶する。配布業者１１４がユーザ要求６２４を受け取ると（直接、または決済機関１２２あるいは他の中間機関を介して）、配布業者１１４は、ユーザ要求６２４、及び権利指定６１４の両方と互換性のあるユーザ許可のセットを作成する（ステップ６２６）。このような互換性のある許可セットがない場合は、ユーザのためのアクションはこれ以上実行されない（オプションでユーザに対し出される通知メッセージを除く）。

【００８９】次にユーザ許可及びユーザの公開鍵６２８を使用し、ユーザが使用できる形式に設定されたカスタマイズＳＰＤ６３２を生成する（ステップ６３０）。ステップ６２６で入手したユーザ許可をＳＰＤ６３２の権利及び許可セグメント５１４に記憶し、ユーザの公開鍵６２８を使用してＳＰＤ６３２のコンテンツセグメント５１６のコンテンツを暗号化する。ここでは公開鍵暗号化機構を使用して、ＳＰＤを汎用形式からカスタマイズＳＰＤ６３２へ変換できる。この機構は、著者、出版社、小売店、顧客等、様々な関係者間で、各段階で権利を保護しながらＳＰＤの機密を守って受け渡しする場合に便利である。さらに、複数のユーザ要求を１つのＳＰＤ６３２内に作成し、格納できることにも注意する必要がある。この技術としては、複数の公開鍵を使用して文書を暗号化し、しかも、任意のユーザ秘密鍵を使用して復号できるような技術が知られている。

【００９０】その結果得られるカスタムＳＰＤ６３２は、コンピュータネットワーク等の利用可能な手段によりユーザ１１８へ送信するか、または物理媒体（磁気ディスクまたは光ディスク等）に格納して頒布される。

【００９１】ユーザがＳＰＤを受け取ったときに実行する操作を図７のフロー図に示してある。まず、ＳＰＤが受け取られ、ユーザシステムに記憶される（ステップ７１０）。通常は、ＳＰＤをただちに使用する必要はない。使用したいとき、通常はユーザ名とパスワードまたは鍵を用いて、最初にユーザの認証が行われる（ステップ７１２）。次に、システムはユーザが希望するアクションを判別する（ステップ７１４）。アクションが選択されると、本発明の権利行使ステップ（ステップ７１６）が実行され、希望するアクションに関連する条件を検査する（料金、時間、アクセスレベル、電子透かし、または他の条件等）。これは、実行可能コードであるＳＰＤアプレット５１２（図５）によりローカルに行うか、または権利実施サーバにアクセスすることで実行できる。

【００９２】権利行使ステップ（ステップ７１６）が失敗すると、更新手順（ステップ７１８）が実行される。ここでユーザは、追加料金を承認するなど、自分の許可を更新する機会が与えられる。条件の検査が正常に終了すると、事前監査手順（ステップ７１８）が実行され、ＳＰＤシステムは検査状態をトラッキングサービス（図1の監査サーバ１３０等）へ記録する。これで、コンテンツは既に説明したように確実に解釈され、画面に表示再生される（ステップ７２２）。ユーザの処理が終了すると、事後監査手順（ステップ７２４）が実行され、使用量がトラッキングサービスにより更新される。そして、ＳＰＤシステムは次のアクションを待つ。

【００９３】ＳＰＤによる保護において特徴的なことは、解釈処理時の中間段階では、ユーザが文書を再配布等の不正利用可能な形式では入手できないようにしていることである。これは、できるだけ後段で、できれば最後のステップで文書コンテンツを復号することで実現されている。

【００９４】図８に、ＳＰＤ復号モデルを示す。Ｅは出版社が実行する暗号機能を示し、Ｄはユーザシステムで実行される復号を示し、Ｒは解釈変換処理を示す。従来システムの多くは最初の変換シーケンスである経路８１０、つまり、Ｄ（Ｅ（ｘ））を実行した後、引き続いてＲ（Ｄ（Ｅ（ｘ））を行っている。既に述べたように、初期の段階で行われる復号では文書は危険な状態に置かれる。できれば、変換は逆順である経路８１２、つまりＲ’（Ｅ（ｘ））を実行した後、引き続いてＤ（Ｒ’（Ｅ（ｘ））を実行するとよい。これにより、復号は可能な限り後段で行われる。

【００９５】Ｒ’が可能かどうか、つまり、復号の前に解釈の処理を実行できるかどうかは、以下の式により判別する。

Ｄ（Ｒ’（Ｅ（ｘ））＝Ｒ（Ｄ（Ｅ（ｘ））

【００９６】ここで、暗号化関数と復号関数とが可換である場合、つまり、任意のｘに対してＥ（Ｄ（ｘ））＝Ｄ（Ｅ（ｘ））となる場合には、Ｒ’が可能かどうかは次の式で確認できる。

ｙ＝Ｅ（ｘ）である時Ｒ’（ｙ）＝Ｅ（Ｒ（Ｄ（ｙ））

【００９７】実際には、ＲＳＡシステム及びＥｌＧａｍａｌ離散対数システム等の一般的な公開鍵暗号システム

の暗号化及び復号関数はこの可換性の要求を満足する。つまり、暗号化及び復号にこれらの復号システムを使用する場合、変換Ｒ’は可能である。

【００９８】パスｘ’＝Ｄ（Ｒ’（Ｅ（ｘ））は、許可されない文書の使用及び配布に対する文書保護の理想的なＳＰＤによる解決方法を示している。文書の配布及び使用のシナリオを以下に説明する。ユーザが文書を購入すると、文書はユーザの公開鍵情報を使用して暗号化し、インターネット等の安全でないネットワークチャネルを介して送信する。暗号化した文書には権利情報が追加されており、権利及び許可を実行する保護アプレット５１２がコンテンツ所有者によりユーザに付与されている。ユーザが文書の使用を要求すると、このアプレットは権利及び許可を確認し、オリジナル文書のプレゼンテーション形式を暗号化文書から生成する。最終的なプレゼンテーションデータ形式になる前の文書の中間形式はどの形式でもユーザの秘密情報により暗号化されているため、文書保護のＳＰＤモデルでは、この文書の中間形式がインターセプトされても他のシステムでは使用できないことが保証されている。

【００９９】この理想的なモデルは、解釈変換処理Ｒに対応する変換処理Ｒ’の計算が効果的に行えるかどうか、特に、Ｒ’実行時に復号関数Ｄを呼び出すことが必要かどうか、に依存していることは明らかである。Ｒ’を効果的に実行できる場合で問題にしなくてもよい自明なケースは、Ｒが暗号化関数Ｅと可換である場合である。この場合、ｙ＝Ｅ（ｘ）について Ｒ’（ｙ）＝Ｅ（Ｒ（Ｄ（ｙ））＝Ｒ（Ｅ（Ｄ（ｙ））＝Ｒ（ｙ）となる。尚、この場合、Ｒ’＝Ｒである。

【０１００】図８から分かることは、２つの極端なケースｘ’＝Ｒ（Ｄ（Ｅ（ｘ））、つまりｘ＝Ｄ（Ｅ（ｘ））に対し保護がない場合とｘ’＝Ｄ（Ｒ’（Ｅ（ｘ））（理想的な保護）との間には、文書保護の問題に対するいくつかの中間的な解決方法（例えば、中間解決方法８１４、８１６、及び８１８）が（上記の想定の下で）存在するということである。図８に示してあるように、暗号化された文書Ｅ（ｘ）からプレゼンテーションデータｘ’を得るには様々なパスがあり、それらは、部分的な解釈変換処理と部分的な復号変換処理が様々に組み合わされたデータに対応していることが分かる。この場合も、どのパスでも復号Ｄを遅らせることで文書の保護レベルが向上することがわかる。

【０１０１】上述のように、復号処理をできるだけ遅らせるという代替的方法では、文書全体や形式ではなく文書のコンテンツだけを暗号化する秘話化技術を採用している。この実現方法を図９に示す。文書コンテンツ９１０は最初は平文である（これは、ユーザ処理時に認識可能な単一の箇所でではなく、図４のステップ４１２実行時に発生する一時的な状態である）。文書は、データ部分９１４と形式部分９１６とに分割される（ステップ９１２）。データ部分９１４は秘話化鍵９２０を使用して秘話化され（ステップ９１８）、平文形式部分９１６とマージされる（ステップ９２２）。これにより、秘話化コンテンツ９２４が得られる。この秘話化コンテンツは、コンテンツを復号しなくても秘話化プレゼンテーションデータに解釈可能である。この秘話化形式の安全性は、秘話化鍵による本格的な暗号化よりも低い。なぜなら、文書のレイアウト、ワードの長さ、行の長さ等から大量の情報が得られ、従って概略の内容が判明してしまうからである。しかし、この方式は、偶発的な著作権侵害を抑止できる。

【０１０２】盲目的変換関数を用いた、再生の際の電子著作物の保護方法は、図１０を参照して示される。図１０において、暗号化電子著作物１０１０が再生用アプリケーションに提供される。電子著作物１０１０は、再生用アプリケーション１０１２が暗号化プレゼンテーションデータ１０１６を生成できるようにする形式保存暗号化方式で暗号化されている。次に、暗号化プレゼンテーションデータ１０１６を復号エンジン１０１８に送り、ここで平文のプレゼンテーションデータ１０２０に復号する。プレゼンテーションデータは今や平文であるが、元のデジタル形式に再び生成されることは恐らくない。ユーザがプレゼンテーションデータ１０２０を直接表示したり使用したりすることができる場合、更なる処理は不要である。しかしながら、プリンタなどの表示システムによっては更なる解釈が必要な場合がある。このような場合、プレゼンテーションデータ１０２０を表示システムの解釈用アプリケーション（プリンタの場合、これは分析装置(decomposer)になりうる）に提供し、このアプリケーションは画像データ１０２４を生成する。画像データ１０２４は次に表示装置１０２６へ提供される。

【０１０３】一般に、盲目的変換の問題を以下のように述べることができる。キャシー(Cathy)というクライアントがサーバーであるスティーヴ(Steve)に、彼の（パブリック又はプライベートな）データａと彼女のプライベートデータを用いて関数値F(a,x)を計算してほしいと希望しており、キャシーは、プライバシー保護の観点から、彼女のプライベートデータｘと関数値F(a,x)をスティーヴに知られずにこの変換が行われることを望んでいる、と仮定する。スティーヴから見れば、これは、キャシーのために目隠しをしたままF(a,x)を計算する、ということである。これは、キャシーがサーバーのスティーヴに、キャシーの鍵ｋで暗号化されたデータ$E_k(x)$のみを用いて変換処理を行い、彼女の鍵ｋを用いて再び暗号化された関数値$E_k(F(a,x))$を彼女に戻してほしいと願っていることを意味する。スティーヴが暗号化データを用いて変換処理を行うことができれば、キャシーはデータｘ及び結果F(a,x)の平文形式での公開を避けることができる。部分的に暗号化されたデータを用いた盲目的変換の理想的なモデルを以下に示す。

【０１０４】
【外２】

$$\begin{array}{ccc} (a,x) & \xrightarrow{E_k} & (a,E(x)) \\ F\downarrow & & \downarrow F' \\ F(a,x) & \xleftarrow[D_{k^{-1}}]{} & F'(a,E(x)) \end{array}$$

【０１０５】この図を可換にする関数F'はスティーヴが実際に計算するもので、変換の結果$F'(a,E_k(x))=E_k(F(a,x))$は所望の関数値F(a,x)を明らかにする復号処理を受ける準備ができている。スティーヴは平文データｘと関数値F(a,x)を「見て」いないため、「盲目的な」変換をキャシーのために行っていることになる。

【０１０６】関数F(a,x)の盲目的評価に関しては、盲目的変換のプロトコルを以下のように説明することができる。

(i)キャシーは暗号鍵kを用いてxを暗号化し、$E_k(x)$を生成する

(ii)キャシーは$E_k(x)$をスティーヴに送る

(iii)スティーヴは平文データa及び暗号化データ$E_k(x)$における関数Fの変換バージョンF'を評価する

(iv)スティーヴは結果$F'(a,E_k(x))$をキャシーに戻す

(v)キャシーは復号鍵k^{-1}を用いて$F'(a,E_k(x))$を復号し、F(a,x)を得る

【０１０７】ここで紹介される盲目的変換の理想的なモデルは、盲目的署名及びインスタンスの隠蔽を一般化したものとしてみなすことができる。盲目的変換は、部分的に暗号化されたデータを入力として許容し、より重要なことには、サーバーが計算する関数F'を目的の関数Fとは異なるものにすることができる。Fの代わりにF'を計算することにより、サーバーは、なお目隠し状態ではあるが、入力が部分的に暗号化されていることを認識し、従ってクライアントに協力することができる。盲目的変換及び安全で融通性のある計算は、サーバが計算する関数値をクライアントの秘密として保つことを共通の目標としているが、盲目的変換ではクライアントがデータ入力を供給するとサーバーが関数（を評価するプログラム）を供給し、安全で融通性のある計算ではその逆である、という点で異なっている。盲目的変換は、データのある部分（例えばa）が平文の形式であることを許容することに注意する。これにより、動的だがなお平文の形式であるいくつかのデータを、表示ウィンドウのサイズ、コンテンツ移動のための参照位置、回転操作における倍率及びスケーリング係数などの解釈処理に使用することができる。

【０１０８】盲目的変換は、暗号化データを計算するための関数F及びF'が可能である場合のみに有効である。加法的暗号化方式を用いた、多変量の整数係数のアフィン関数により、x座標及びy座標上のアフィンタイプの文書解釈関数の多くを盲目的変換で評価できることを示すことができる。与えられた暗号化方式Sに対し、ある関数F':X→Xが可能である場合、関数F:X→Xは「Sによる盲目的計算が可能」と言われる。これにより、F'を評価するための計算の複雑性はFを評価するための計算の複雑性の多項式となり、任意の$k\in K$及び$x\in X$に対して$F(a,x)=D^{k-1}(F'(a,E_k(x)))$となる。暗号化方式Sが可能である場合、関数F:X→Xは「盲目的計算が可能」と言われ、SによるFの盲目的計算が可能であるようにXはメッセージスペースの部分集合である。

【０１０９】多変量の整数係数のアフィン関数はいずれも、任意の加法的暗号化方式に対してSによる盲目的計算が可能である。

【外３】

具体的には、$F_{x0;a1,\ldots ak}(x_1,\ldots,x_k)=x_0+\sum_{i=1}^{k}a_ix_i$ を、定数 $x_0\in X$、整数係数 a_i、及び X の変数 $x_1,\ldots x_k$ を有する多変量アフィン関数とする。任意の鍵 $k\in K$ に対して計算効率の高い関数 $F'_{y_0,b_1,\ldots b_k}(y_1,\ldots,y_k)=y_0\oplus\bigoplus_{i=1}^{k}b_iy_i$ が可能となり、これにより、

$$E_k(F_{x_0,a_1,\ldots a_k}(x_1,\ldots,x_k))=E_k(x_0+\sum_{i=1}^{k}a_ix_i)=F'_{y_0,b_1,\ldots b_k}(E_k(x_k))$$ となる。

実に、$F'_{y0,b1,\cdots,bk}$の定数y_0及び整数係数b_iを取り入れて$y_0=E_k(x_0)$，$b_i=a_i$，i=1,…,kとすることができる。本明細書に記載の、文書の形式保存暗号化及び高信頼性解釈の理論上の基礎を条件とする場合、加法的暗号化方式を用いた多変量の整数係数アフィン関数の盲目的変換により、x及びy座標上のアフィンタイプの文書解釈関数の多くを盲目的に評価することができる。

【０１１０】文書は、通常は一定の形式にならったメッセージである。文書を暗号化するには、文書全体を単に暗号化する他に、文書のある部分のみを暗号化する多くの様々な方法がある。本明細書での目標は、暗号化されていない部分に関する情報の漏洩を使用できないようにすること、即ち、情報が漏洩しても平文のオリジナル文書の再構築を計算上困難なものにすることである。

【０１１１】暗号化方式が電子著作物の形式情報を保つ場合、任意の変換関数（再生用アプリケーション又は解釈用アプリケーション）を使用することができる。形式保存暗号化方法の一例を、便宜上トークンベースの文書を参照して説明する。形式保存暗号化の方法を、他の形式（例えばHTML/XML、Microsoft WORD、Acrobat PDFなど）の文書に容易に拡張又は適用することができる。Xerox社の DigiPaperのようなトークンベース形式では、文書の各ページ画像はトークン画像（文字及びグラフィックエレメントなど）の「辞書」及び位置情報（トーク

ン画像がページのどこに現れるかを示す）として表される。従って、文書中に同一のトークンが複数回現れても、辞書内のそのトークンの画像を１つだけ用いて表すことが可能である。

【０１１２】このような形式で文書を解釈する方法は、トークンの位置を連続的に読み取り、トークンの画像を辞書から取り出し、そして指定の位置で画像を描くことによって達成される。トークンベースの文書の利点は、ファイルサイズがコンパクトであり、電子文書の配布、表示及び印刷に使用する際の解釈速度が速いことである。DigiPaper形式では、トークンはCCITT グループ4圧縮形式を用いてバイナリ画像として記憶されるか又はJPEG圧縮を用いてカラー画像として記憶され、トークンの位置情報はハフマン符号を用いて更に圧縮される。

【０１１３】便宜上、P枚のページからなるトークンベースの文書Dを、P枚のページの画像を表すサイズ$|L_k|$の位置L_kのテーブルP個のシーケンス$(1 \leq i \leq P)$と共に、サイズ$|T|$のトークンTのテーブル（辞書）として正式にモデル化する。各エントリT[j]（$1 \leq j \leq |T|$）は、j番目のトークンの識別子id[j]及び画像t[j]を有する組(id[j], t[j])である。i番目の画像位置テーブルL_iにおける各エントリL_i[k]（$1 \leq k \leq |L_i|$）は、i番目のページ画像におけるk番目のトークンの発生を表す組(id[k],x[k],y[k])であり、ここでid[k]は識別子、そしてx[k]及びy[k]はページ内の前の(k-1)番目のトークン発生からのx座標及びy座標の差である。例えば、図１１に示す簡潔な文書を考える。この文書のトークン辞書及び（x、y座標を用いた）位置テーブルをそれぞれ図１２及び図１３に示す。

【０１１４】下記の概略的な擬似コードRender(D)は、文書Dのページ画像を解釈する態様を示している。このコードにおいて、x_0, y_0は各ページ毎のx座標及びy座標のベース参照であり、Lookup(T,id[k])は、辞書T及びトークン識別子id[k]が入力されると所与の識別子に対応する辞書T内のトークン画像に戻るサブルーチンであり、Draw(x,y,t)は位置(x,y)でトークン画像tを描くサブルーチンである。

【０１１５】
【外４】

```
Render(D)
{
    Load T into memory
    for i = 1 to P do
    {
        Load Li into memory
        x = x0
        y = y0
        for k = 1 to |Li| do
        {
            x = x + x[k]
            y = y + y[k]
            t = Lookup(T,id[k])
            Draw(x,y,t)
        }
    }
}
```

【０１１６】前述の概略的な解釈方法に使用されるようなシフト変換x'=x+a, y'=y+bの他に、文書の解釈の際に生じうるいくつかの他の座標変換がある。

【０１１７】スケーリング

スケーリング変換はx'=ax, y'=byという形からなり、ここでa及びbはそれぞれx座標及びy座標の倍率である。スケーリングは、表示ウィンドウ又は印刷用紙をリサイズすることによって生じうる。

【０１１８】回転

【外５】

回転変換は、２×２の回転マトリックスを形成する定数a、b、c、dに対して

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$ である。この変換は、ページ画像を回転させる際に必要である。

【０１１９】アフィン変換

アフィン変換は、いくつかの定数a、b、c、d、e、fに対してx=ax+by+e、 y=cx+dy+fの形をとるものである。

【外６】

ベクトルの形では、$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$ である。

シフト変換、スケーリング変換及び回転変換がアフィン変換の特殊なケースであることは明らかである。後述する加法的暗号化方式を用いた座標情報の暗号化において高レベルの信頼性の解釈を達成することができるのは、これらのアフィンタイプの変換である。

【０１２０】特殊な類の暗号化方式、即ち加法的暗号化方式は、文書の高信頼性解釈の基礎を提供するアフィンタイプの関数の盲目的変換を実行するために用いられ

る。暗号化文書の解釈変換R及びR'による盲目的変換は、D(R'(E(x)))=R(D(E(x)))の関係を満たす。式中、Eは暗号化関数で、DはEのための復号関数である。E(x)が加法的暗号化方式である場合、R'=Rである。

【０１２１】暗号化方式Sは一般に、(i)可能なメッセージの集まりであるメッセージスペースX、(ii)可能な暗号化メッセージの集まりである暗号化テキストスペースY、(iii)可能な鍵の集合である鍵スペースK、(iv)計算上効率的な暗号化関数E:K×X→Y、及び(v)計算上効率的な復号関数D:K×Y→X、といった基本的に５つの構成要素からなる。各鍵k∈K毎に固有の鍵k^{-1}∈Kがあり、これにより、暗号化関数$E_k=E(k,):X\rightarrow Y$及び復号関数$D_{k-1}=D(k^{-1},):Y\rightarrow X$は、各メッセージx∈X毎に$D_{k-1}(E_k(x))=x$を満たす。鍵kは暗号鍵と呼ばれ、$k^{-1}$はこれに対応する復号鍵と呼ばれる。

【０１２２】このように定義された暗号化方式をいくつかの方法で変更し、実際に用いられる広範囲の具体的な暗号化方式をカバーすることができる。１つの変形例は、暗号化及び復号に用いる鍵が互いに異なるか否かを考慮することである。全ての暗号鍵kが対応する復号鍵k^{-1}と同一である場合、この方式は対称的（又は秘密鍵）であり、そうでない場合、この方式は非対称的である。全ての可能なkに関して、k^{-1}がkとは異なり、kからの導出が計算上困難である場合、この方式は公開鍵暗号化方式である。

【０１２３】別の変形例は、決定的暗号化方式と確率的暗号化方式とを区別することである。決定的方式では、全ての暗号化関数E_k及び復号関数D_{k-1}は決定的な関数であるが、確率的方式では暗号化関数E_kを非決定的とすることができ、即ちこの関数をメッセージに２度適用することによって２つの異なる暗号化メッセージを生成することができる。

【０１２４】加法的な暗号化方式とは、メッセージスペースX及び暗号化テキストスペースYがある加法的構造を有し、暗号化関数$E_k=E(k,):X\rightarrow Y$が加法的構造に対して準同型である暗号化方式である。

【外７】

具体的には、X=(X, +, 0)及びY=(Y, ⊕, 0)を、例えば全てのxに対してx+0=x及び0+x=xを満たす（異なりうる）ゼロエレメント0と、有効演算子+及び⊕とを有する２つの可換半群(commutative semigroups)とする。任意のk∈K及び任意のx, x'∈Xに対して$E_k(x+x')=E_k(x)\oplus E_k(x')$であり、演算子⊕が平文のメッセージx及びx'を明らかにしない場合、暗号化方式は「加法的」であると言われる。⊕に対する最後の条件により、加法的暗号化方式は自明ではなくなる。この条件がない場合、Yに対する演算子⊕を$y\oplus y'=E_k(D_{k-1}(y)+D_{k-1}(y'))$として自明に定義することができる。即ち、この定義は、初めに引数を復号し、これらの引数を互いに加算して、最後に結果を再び暗号化することによって達成される。

【０１２５】

【外８】

加法的暗号化方式に密接に関連しているのは、乗算暗号化方式である。スペースX及びYが環構造（即ち、加法的構造に加え、加法+及び⊕に対して分配される各々の乗算×及び⊗、並びに乗算恒等式を有する）を有し、暗号化関数E_kが乗算に対して準同型であり、$E_k(x\times x')=E_k(x)\otimes E_k(x')$であり、演算子⊗が平文のメッセージx及びx'を明らかにしない場合、暗号化方式は「乗算的」であると言われる。

【０１２６】一般に、加法的（及び乗算）暗号化方式は非順応的ではない。なぜなら、暗号化メッセージが与えられると、各々の平文メッセージが関連付けられるような異なる暗号化メッセージの生成が（少なくとも計算上は）不可能である、ということが、非順応性の方式には必要とされるからである。従って、加法的暗号化方式は、攻撃者が暗号化メッセージを他の方法で削除、追加、又は変更しようとするアクティブな攻撃に対して弱点を有する。しかしながら、これらの方式を文書の暗号化に用いた場合、文書の完全性及び機密性に対するこれらのアクティブな攻撃によって生じるリスクを減少させるように、データの完全性及びメッセージの認証に対して更なる手段を講じることができる。更に、これらの攻撃は、ユーザが使用及び消費しようとする文書のコンテンツに影響を及ぼすため、エンドユーザの、アクティブな攻撃を開始しようとする動機は薄くなる。

【０１２７】全ての暗号化方式を加法的なものであると、容易に、そして当然のこととして定義することはで

きない。実際に、非加法的であるか又は少なくとも非加法的なものに変換可能であることを要件として設計された暗号化方式もある。しかしながら、形式保存暗号化及び高信頼性の文書解釈の方法に用いることのできる付加的暗号化方式の例はたくさんある。Mult、Exp及びEG（３つの決定的方式）、OU（確率的）、並びにRSAは、形式保存方法に用いることのできる付加的暗号化方式の例である（攻撃を受ける程度は様々である）。

【０１２８】乗算暗号(Mult)は対称的暗号化方式であり、ある整数n＞0に対してX=Y=Z_n={0,1,…n-1}である。鍵aを用いたメッセージxの暗号化は$y=E_a(x)=ax \pmod n$であり、鍵aを用いたメッセージyの復号は$x=D_a(y)=a^{-1}y \pmod n$であり、式中a^{-1}はモジューロnの乗算逆数である。

【０１２９】指数関数暗号(Exp)は対称的暗号であり、ある素数pに対してX=Z_{p-1}及び暗号テキストスペースY=Z_pであり、Kは乗算群Z^*_pの全ての生成元の集合である。任意の生成元g∈Kに対して、暗号化関数は指数関数$E_g(x)=g^x \pmod p$として定義され、復号関数は対数関数$D_g(y)=\log_g y \pmod{(p-1)}$として定義される。

【０１３０】準確率的ElGamal暗号(EG)は、指数関数暗号をElGamal暗号に拡張しており、ElGamal暗号を準確率的なモードで実行する。各メッセージに対してx∈Z_p（ある素数pに対してZ_p={1,…p-1}）であり、gは乗算群Z^*_p内の生成元であり、ユーザ用の秘密復号鍵は乱数a∈Z^*_{p-1}であり、公開暗号鍵は$\alpha=g^a \pmod p \in Zp$であり、暗号化$E\alpha(x,r)$は一様に選択される乱数r∈$Z^*_{p-1}$に依存する。即ち、$E\alpha(x,r)=(g^r \pmod p, x\alpha^r \pmod p)=(s,t)$である。暗号化メッセージ(s,t)に対し、復号関数は$D\alpha(s,t)=t(s\alpha)^{-1} \pmod p$である。

【０１３１】

【外９】

前述のようなオリジナル形式のElGamal暗号はほとんど加法的ではない。しかしながら、下記のように、同一の乱数ｒを共有するｘ'の暗号テキスト上で演算子⊕を部分的に定義することができる。

$$E_\alpha(x,r) \oplus E_\alpha(x',r)=(s,t) \oplus (s,t')=(s,t+t')=E_\alpha(x+x' \pmod p, r)$$

この部分的に定義された演算は、メッセージのバッチが同一の乱数ｒを用いて暗号化される際に適用可能である。

【０１３２】オカモト－ウチヤマ(Okamoto-Uchiyama)暗号(OU)

オカモトとウチヤマは、T.オカモト及びS.ウチヤマの"A New Public-Key Cryptosystem as Secure as Factoring"(Eurocrypt'98, Lecture Notes in ComputerScience 1403, 308-318, 1998)において、加法的な公開鍵暗号化方式を提唱した。この方式は確率的であり、おそらく消極的な攻撃者に対して$n=p^2q$のファクタリングの処理が困難であるのと同じ位安全度が高いであろう。k＞0であるk個のビットのうち２つの大きな素数p、qを選択し、$n=p^2q$とする。$g_p=g^{p-1} \pmod{p^2}$の位数がpになるように、g∈Z^*_nをランダムに選択する。$h=g^n \pmod n$とする。OU方式のメッセージスペースXは（オカモト及びウチヤマが主張するような集合{1,…2^{k-1}}ではなく）集合Z^*_pであり、暗号テキストスペースYはZ_nである。ユーザに対しては、公開鍵は組(n,g,h,k)であり、対応する秘密鍵は素数の対(p,q)である。メッセージx∈Xを暗号化するには、乱数r∈Z_nを一様に選択する。すると、暗号化メッセージは$y=E_{(n,g,h,k)}(x,r)=g^x h^r \pmod n$となる。暗号化メッセージyを復号するには、「対数」関数L:Γ→Γ、$L(x)=(x-1)p^{-1} \pmod{p^2}$を用いる。式中、Γは$Z^*_{p^2}$のp-Sylow半群であり、即ち、Γ={x∈$Z^*_{p^2}$|x≡1(mod p)}である。関数Lを用いた場合、復号関数は$x=D_{p,q}(y)=L(y^{p-1} \pmod{p^2})L(g_p)^{-1} \pmod{p^2}$である。

【０１３３】新しい加法的暗号化方式を、暗号化方式の合成構築によって既存のものから構築することができる。また、非加法的暗号化方式から加法的暗号化方式を構築する際にも合成構築を用いることができる。例えば、指数関数暗号Exp及び任意の乗算暗号化方式S（RSAなど）の合成によって加法的暗号化方式が生じる。

【０１３４】前述のように、文書の高信頼性解釈の基礎として機能する、部分的に暗号化されたデータを用いる盲目的変換は、加法的暗号化方式によって可能になる。特に、加法的暗号化方式を使用して、平文係数及び暗号化変数を用いたアフィン関数の盲目的変換を行うことができる。

【０１３５】トークンベースの文書の例に戻ると、トークンベースの文書Dはトークン画像の辞書T及び位置テーブルL_i（各ページ画像につき１つ）のシーケンスから構成されるため、辞書T及び位置テーブルL_iのコンテンツを暗号化し、暗号化されたトークン画像の辞書T'、及び暗号化された位置のテーブルL'_iを生成することを意図する。辞書Tは、対(id[j],t[j])(j=1,…|T|)の集まりからなることを想起する。Tに関連づけされているのは、有効なトークン識別子idが与えられるとT内の対応トークン画像tを返す、解釈処理のサブルーチンLookupである。辞書Tを暗号化する際、トークン識別子の暗号化、トークン画像の暗号化、又は両方といった３つの基本的な選択肢がある。識別子又はトークン画像を暗号化することにより、識別子とそのトークン画像との間のつながりをアンリンクする手助けをする。更に、トークン画像を暗号化することによって所有権のあるトークン画像が

保護される。いずれにせよ、解釈処理Pのみでの辞書への有効なアクセスを許容する一方、辞書の平文コンテンツ全体のコピーを得ることを計算上難しくすることが望ましい。このことは可能である。なぜなら、多くの場合、有効な識別子（例えばハフマン符号語）は、全てが特定長のバイナリストリングからなる非常に小さな部分集合にすぎず、結果的に、識別子の全数探索はどれも効率的ではないからである。

【０１３６】より形式的には、辞書T及びこれにアクセスするLookupサブルーチンが与えられた場合、辞書の暗号化の要件は、暗号化された辞書T'及び対応するサブルーチンLookup'が下記の制約を満たすことである。

(1)任意の暗号化識別子$E_k(id)$に対し、Lookup'(T', $E_k(id)$)= E_k(Lookup(T,id))

(2)T'及びLookup'を与えられてもTの再構築が計算上不可能である

【０１３７】暗号化方式Sに対し、T'及びLookup'を以下のように構築することができる。IDを構文的に可能な全ての識別子の集合とし、特に、$ID^* \subseteq ID$（式中、$ID^* = \{id \mid (id,t) \in T\}$）とする。hを、ドメインがIDである一方向ハッシュ関数とする。次に、T内の各対(id,t)毎に、対$(h(id), E_k(t))$をT'に挿入することにより、暗号化トークン辞書T'をTから取り出す。変換されたサブルーチンLookup'は下記のアルゴリズムを用いる。

【外１０】

```
Lookup'(T',id)
{
        id' = h(id)
        t' = Lookup(T',id')
        return (t')
}
```

Lookup'の戻り値は暗号化トークン画像であることに注意する。この画像の復号は、後述する高信頼性解釈の一部である解釈処理の最終サブルーチンDraw'まで延ばされる。

【０１３８】この辞書の暗号化は、記憶スペースオーバーヘッド及び実行時オーバーヘッド双方の点で、トークン辞書の暗号化版を用いた計算を、計算上実行することができる。Lookup'サブルーチンで用いたハッシュアルゴリズム及び暗号化アルゴリズムが安全性の十分に高いものである場合、T'及びLookup'を与えられてもTの回復は計算上非常に困難である。

【０１３９】位置テーブルL_iへの各エントリは、識別子、並びにx座標及びy座標における位置の差からなるため、これら３つの要素のどの組み合わせでも暗号化することができる。位置情報を暗号化するには、アフィンタイプのあらゆる解釈変換を位置座標に適用できるように、加法的暗号化方式が推奨される。識別子に関しては、文書圧縮と文書保護との間のトレードオフ（妥協点）を決めなくてはならない。トークンベースの文書では、トークン識別子は通常、圧縮を目的とするある符号化方式の符号語である。例えば、ハフマン符号を文書の圧縮に用いる場合、識別子は、文書内の発生回数に基づいた、トークンのバイナリのハフマン符号語である。この場合、これらの識別子を暗号化するために単に決定的暗号化方式を用いても、識別子は有効に保護されない。これは、この方式が各トークンの発生回数を変えないため、誰でも暗号化識別子の発生回数を再カウントでき、識別子であるハフマン符号語を再構築できてしまうからである。従って、文書内のトークンの発生回数を隠すためには、確率的暗号化方式を用いて識別子を暗号化するのが好ましい。しかしながら、この暗号化により、識別子（符号語）に保持される最適な符号化が妨げられ、文書の圧縮比が小さくなってしまう。良好な文書圧縮の達成はトークンベースの文書の設計目標の１つであるため、これはトークンベースの文書には望ましくない場合がある。

【０１４０】L_iを暗号化するための穏当な妥協案を提案する。暗号化及び復号の効率が大きな問題でない場合は、加法的暗号化方式S、好ましくは、オカモト-ウチヤマ暗号OUのように確率的で非対称のものを選択する。L_iへの各エントリ(id,x,y)毎に、$(id, E_k(x), E_k(y))$をL'_iに挿入する。識別子の暗号化も必要である場合は、$(E_k(id), E_k(x), E_k(y))$のようなエントリを位置テーブル$L'_i$に挿入することができる。しかしながら、この場合、暗号化辞書T'へのエントリを$(E_k(id), E_k(t))$に変える必要があり、前述のサブルーチンLookup'も、この変更を反映するように修正する必要がある。

【０１４１】前述のトークンベースの文書の形式保存暗号化を用いて、解釈処理の際に文書のコンテンツも保護することができる。Draw'(x,y,t)への暗号化を遅らせることを意図とする。解釈処理は、下記に示されるようなものである。

【０１４２】

【外１１】

```
Render(D)
{
        Load T into memory
for i = 1 to P do
{
        Load Li into memory
        x = Ek(x0)
        y = Ek(y0)
        for k = 1 to |L| do
        {
                x = x ⊕ x[k]
                y = y ⊕ y[k]
                t = Lookup'(T',id[k])
                Draw'(x,y,t)
                }
        }
}
Draw'(x,y,t)
{
        x = Dk-1(x)
        y = Dk-1(y)
        t = Dk-1(t)
        Draw(x,y,t)
}
```

この処理の際、サブルーチンDraw'(x,y,t)を呼び出す前は、座標及びトークン画像の情報は全て暗号化されたままである。この暗号化方式は加法的であるため、これは座標情報に対して可能である。結果的に、解釈処理のコンテンツ保護レベル及び解釈処理の性能は、用いる方式の秘密保持強度及び計算の複雑性に依存する。

【０１４３】本発明の他の実施形態では、電子コンテンツ又はプレゼンテーションデータを秘話解除せずに電子著作物を高信頼性で解釈又は再生することができるように、電子著作物を秘話化する。この実施形態では、電子著作物は、電子コンテンツ及びリソース情報（システムコンテクストとも呼ばれる）を含むタイプのものである。リソース情報は、電子著作物をプレゼンテーションデータに変換するために再生用又は解釈用アプリケーションが用いる形式情報又は他の情報を含む。

【０１４４】秘話化は、元のコンテンツを読み取り不能又は使用不能にする変換処理の一種である。電子著作物wに対しては、シードsを用いる秘話化方式Tが秘話化電子著作物w'をw'=T(w,s)に従って生成する。同一の変換Tを用い、秘話化リソース情報S'をS'=T(S,s)に従って生成することもできる。この例では、秘話化方式のリバース・エンジニアリングをより難しくするためにシードsを用いる。

【０１４５】例えば(For example)、簡潔な秘話化方式を用いて文書タイプの電子著作物を秘話化することができる。文書において、電子コンテンツは一連の文字を特定の順序で又は特定の位置に含んでいる。この文書を表示装置上で表示する場合、各文字を特定の位置に表示して、ユーザがモニタなどの表示装置で眺めることができるようにしなくてはならない。各文字をモニタに表示するには座標系が必要であり、これによって文書の各文字をモニタに表示することができる。電子コンテンツは座標情報を含み、この情報はモニタの座標系によって参照される。例えば、この段落では、"F"の文字が最上行に現れており、行頭より５字分だけ下げられている。

【０１４６】上の段落のテキストを無秩序(jumbling)にするための簡潔な秘話化方式は、文字の位置を座標系に対して移動させることである。段落内の各文字は(x,y)の位置を有する。上の段落の各文字の位置(x,y)を、ユーザシステムからのシード(a,b)を用いて秘話化すると想定する。下記の秘話化関数を用いて、上の段落を秘話化することができる。

縦軸についてY=b^y

横軸についてX=x/a

【０１４７】この例において、再生用アプリケーションが電子コンテンツをプレゼンテーションデータに変換するには、即ち段落をスクランブルしていない状態でモニタに表示するには、ユーザの装置の座標系を秘話化しなくてはならない。秘話化座標系を生成するには、ユーザの装置の座標系を、同一のシード(a,b)を用いて秘話化しなくてはならない。下記の変換関数を用いて、所与の点のx及びy双方の位置を計算する。

縦軸についてY=$\log_b(Y)$（$\log_b$は底bを有する対数）

横軸についてX=aX

【０１４８】再生用アプリケーションが秘話化された電子著作物における文字の位置を得る際、この位置は(X,Y)=(x/a,b^y)によって与えられる。そして、この値を装置の座標系に適用し、(X,Y)=($\log_b(Y)$,aX)=(x,y)とする。"F"の正確な位置はこのようにしてユーザのモニタに表示される。双方の秘話化の場合において、リソース情報及び電子著作物の秘話化形式は固有の連関を保っている。リソース情報及び電子著作物の、これらの相補的な秘話化形式により、電子著作物を保護する有効なメカニズムの基礎が生じる。再生用アプリケーションは秘話化された電子著作物を表示することはできるが、再生用

アプリケーションは、秘話化されたシステムのコンテクストを用いてのみ、平文のプレゼンテーションデータを提供することができる。

【０１４９】一般に、秘話化は暗号化ほど保護が厳密ではないが、保護する電子著作物の重要性により、異なるレベルの秘話化を用いることができる。重要性の高い著作物は高レベルの秘話化を必要とし、重要性の値がより低い著作物はより弱いタイプの秘話化を必要としうる。ユーザの環境が高信頼性のものである場合、低レベルの秘話化を用いることができる。低レベルの秘話化の使用に際する有利な点は、秘話化電子著作物の作成や秘話化電子著作物の解釈又は再生に必要とするシステムリソースの数が少なくてよいことにある。また、秘話化シードのタイプ及び品質を秘話化方式と組み合わせて用い、秘話化のレベル及び強度を決定することもできる。例えば、より複雑な秘話化シード（高信頼性ソースからの許可情報を含むもの、又は動的シードなど）は、より高レベルの秘話化及び強度をもたらすであろう。

【０１５０】秘話化は一般に、配布場所又は製造場所において生じる。電子著作物は通常、ユーザ又はカスタマに配布される前に、製造者又は配布業者が選択した秘話化方式を用いて秘話化される。秘話化されるリソース情報も、配送前に予め選択することができる。各秘話化方式毎にシードを用いることが好ましい。また、ユーザシステムのコンテクストによって提供される情報を用いてシードを生成することが好ましい。

【０１５１】ユーザが電子著作物を購入する際に、ユーザは、その電子著作物の再生のために指定するユーザシステムからの情報を提供することが好ましい。この情報を用いて、秘話化電子著作物及び秘話化リソース情報（秘話化システムコンテクストと呼ばれることもある）双方の秘話化シードを生成することができる。それから、秘話化電子著作物及び秘話化システムコンテクスト又は秘話化リソース情報をユーザに提供する。また、本発明のこの実施形態の作用では不要であるが、一般に、秘話化電子著作物及び秘話化システムコンテクストを、ユーザへの配布前に暗号化することができる。使用する復号方式によっては、秘話化電子著作物及び秘話化システムコンテクスト双方の復号を、秘話化電子著作物のプレゼンテーションデータへの再生前に行わなくてはならない場合がある。

【０１５２】秘話化電子著作物の生成方法は３つのステップに分けられる。これらのステップは、秘話化シードの生成、電子著作物の秘話化、及びリソース情報の秘話化である。秘話化シードが生成されると、秘話化エンジンに秘話化シードが与えられる。秘話化エンジンは、電子著作物又はリソース情報を入力として用い、秘話化シードが与えられた変換関数に基づいて、電子著作物又はリソース情報の秘話化形式を生成する。秘話化電子著作物の再生の際は、秘話化リソース情報を用いてプレゼンテーションデータ及び/又は画像データを生成する。同一の又は異なる秘話化変換関数を、電子著作物及びリソース情報に対して使用することができる。

【０１５３】秘話化電子著作物の生成方法は、図１４に関連して示される。電子著作物１４１０は、電子コンテンツと、ユーザが使用可能又は表示可能な形式に電子コンテンツをフォーマットし、解釈するのに用いられるリソース情報のセットとを含む。電子著作物１４１０はコンテンツ秘話化１４２０の処理を受け、この処理において電子コンテンツを秘話化してリソース情報を保存し、秘話化電子著作物１４２２を生成する。コンテンツ秘話化１４２０を、図９に関連して示すように生じることができる。電子著作物は一般に、コンテンツ、命令、及びフォーマットを含む。電子著作物全体を秘話化することができるが、コンテンツのみを秘話化し、命令及びフォーマットを秘話化しないことが好ましい。しかしながら、いくつかの場合では、電子著作物に含まれるリソース情報の一部をいくつかの再生用アプリケーションのために秘話化することもできる。これは、前述の形式保存暗号化方式に関しても同様である。

【０１５４】リソース抽出１４１２は、電子著作物１４１０に関連するリソース情報のセットから少なくとも１つのリソース情報を抽出する。抽出は、リソース情報をシステムリソースファイル１４１４にコピーすることによりなされる。それから、リソース秘話化１４１６においてシステムリソース１４１４を秘話化し、秘話化システムリソース１４２４にする。コンテンツ秘話化及びリソース秘話化のための秘話化方式を同一にする必要はない。各秘話化方式は、シード生成装置１４２６によって生成される秘話化シード１４１８を用いることが好ましい。例示的なシード生成方法をいくつか後述する。特に、好適な実施形態では、秘話化シードはユーザシステムからの固有の情報に基づいている。

【０１５５】秘話化シードの生成技術をいくつか使用することができる。例えば、乱数生成装置から数を生成するシード生成装置を用いることができる。無状態秘話化と呼ばれるこの方法は、秘密鍵情報及びユーザシステム情報のいずれにも依存しない。無状態秘話化の方法は、秘話化用のシステムに対して特定の値を生じる。電子セキュリティシステム固有の弱点が、秘密情報の誤った取り扱い、数学的な複雑性、及びアルゴリズムの複雑性にみられる場合がある。秘密情報を取り除くことにより、攻撃の的が１つ減る。無状態秘話化により、乱数生成装置は秘話化シードを生成する。この場合、秘話化処理が完了すると、シードは跡を残さずに廃棄される。従って、システムの安全性は秘密情報の漏洩に集中した攻撃を免れ、ユーザはプライバシーの侵害とされうる重要な情報を漏らす必要がなくなる。

【０１５６】使用することのできる他のシード生成装置は、状態ベースの生成装置である。状態ベースのシード

生成装置は、まずシステムの状態の情報をユーザの再生システム又は解釈装置から取得することによってシードを構築する。システムの状態の情報は、ハードウェア識別子、システム設定、及び他のシステム関連情報を含む。無状態秘話化の評価は高いが、他の安全性の要件により、特定のユーザシステム又は装置への不可分なリンクを使用しなくてはならない場合がある。システム／装置に特有の情報から秘話化シードを生成することにより、秘話化エンジンは、特定のシステム／装置に対応する形式に秘話化された電子著作物を生成する。

【０１５７】秘話化シード生成装置を許可処理に関連させることもできる。許可ベースの秘話化では、シードの生成を許可処理の結果に関連させることができる。（高信頼性ソースである）別個の許可リポジトリは、電子著作物へのアクセス権をユーザに配送することに関連する他の安全性機能の一部としての許可情報を提供する。許可情報の高信頼性ソースを、米国特許第５，６２９，９８０号に記載のようなオンライン許可リポジトリとすることができる。そして、この許可情報を使用して秘話化シードを生成する。

【０１５８】無状態秘話化シードを用いる場合、電子著作物及びそのリソース情報を秘話化し、一緒に記憶して、ユーザが特定の電子著作物に関連する使用権利を購入した際にユーザに送ることができる。これらの他の秘話化シード生成方法のうちの１つを使用する場合、一般に、ユーザがシステム情報又は許可情報を提供するまで秘話化の実行を待たなくてはならず、その後に電子著作物及びリソース情報を秘話化することができる。

【０１５９】電子著作物を特定の物理システム又は装置で再生可能にすることを確実にするという点でより高度なレベルの保護を提供する実施形態は、動的状態ベースの秘話化シードを用いる。この実施形態では、電子著作物及びリソースの情報と共に秘話化エンジン及び秘話化シード生成装置を再生用アプリケーション又は解釈装置に提供しなくてはならない。この実施形態では、再生及び解釈前に、特定のシステム又は装置の動的状態に基づいて生成されたシードを用いて電子著作物及びリソースの情報を秘話化する。動的状態は、例えばシステムクロック、ＣＰＵの利用、ハードドライブの配置、カーソルの座標などから生じうる。動的状態のスナップショットを用いて著作物を秘話化することにより、著作物は特定のシステム構成（即ち、状態）に調子を合わせてロックされる。電子著作物の秘話化、そして最終的にその盲目的再生（後述）は、動的進展状態に基づいている。動的状態の進展は、秘話化処理の再現を可能にする固有の秘密情報を生じないため、動的状態ベースの秘話化により、秘話化された電子著作物及びシステムコンテクストの漏洩が難しくなる。秘話化処理は高信頼性システム内で行われるため、このことは、この処理の構築を解くことが不可能であることを意味している。

【０１６０】前の例で説明したように、秘話化の実際の処理を、秘話化シードによってアルゴリズムベースの変換でパラメータ化することができる。秘話化の際、電子著作物のデータ及びリソース識別子を前述のように変換する。しかしながら、電子著作物の構造は変わらず、PDF、DOC、WAV、又は他の形式などの元の形式は、形式保存暗号化の際と同様に維持される。同様に、リソース情報の秘話化によってリソース情報の秘話化形式が生じ、これによってリソース識別子、エレメント識別子及びリソース特性は変換されるが、システムコンテクストの構造は変わらないままになる。ユーザの特定の装置又はシステム情報に基づいて、電子著作物及びリソースの情報を同一シードによって秘話化することによって不可分の関係が確立され、いずれの他の装置又はユーザシステムを用いても著作物をその平文の形式に再生することができなくなる。たとえ未許可で配布されても、保護は有効のままである。

【０１６１】盲目的再生の際、秘話化リソース情報の固有の特性により、再生用アプリケーションは秘話化電子著作物を適切に再生し、秘話化されていない、即ち平文のプレゼンテーションデータを生成することができる。電子著作物及びリソース情報を相補的に変換したため、リソース識別子及びデータなどの電子著作物の秘話化エレメントは、システムコンテクストのリソース内の相補エレメントを、内容を知らないまま参照する。照合変換により、コンテクスト内の適切なエレメントは再生用アプリケーションによって識別され、これによって生じるプレゼンテーションデータが平文で現れる。従って、著作物は再生後できる限り後段まで保護される。

【０１６２】前述したように、電子著作物のウェブを介する従来の配布は比較的簡単である。著作物は、エディタを用いて作成され、ウェブサイトに掲載され、ユーザ読者によってアクセスされ、表示装置又は表示システムで再生される。コンテンツ所有者が自分の電子著作物の保護を望まない場合（又はコンテンツ所有者が著作物を受け取るユーザ全員を信頼する場合）、電子著作物は「平文で」、即ち符号化、暗号化又は他の保護を全くせず、どのユーザでも直接使用することができる態様で提供される。

【０１６３】電子著作物がユーザシステムにダウンロードされる場合、電子著作物は一般にメモリに記憶される。電子著作物が、フロッピー（登録商標）ディスク、CD-ROM又はDVD-ROMなどの記憶媒体を介して提供される場合、電子著作物は通常記憶媒体から直接アクセスされる。

【０１６４】図１５を参照すると、電子著作物を再生するために、電子著作物１５１０を再生用アプリケーション１５１２に提供する。形式情報又はリソース情報を必要とする文書又は他のタイプの著作物の場合、電子著作物は、電子コンテンツと、再生用アプリケーションが電

子コンテンツを処理するのに必要な特定のシステムコンテクスト又はシステムリソースを示すリソース情報と、を含む。例えば、電子著作物１５１０を、テキストがArialフォントを用いて表示されるテキスト文書とすることができる。再生用アプリケーション１５１２が、Arialフォントを用いていることを示す電子著作物１５１０のリソース情報にアクセスする際、再生用アプリケーション１５１２は適切なシステムリソース１５１６（この場合はArialフォントテーブル）にアクセスし、システムリソース情報を用いて電子コンテンツをプレゼンテーションデータ１５１４に変換する。

【０１６５】いくつかの再生用アプリケーションでは、電子コンテンツのプレゼンテーションデータへの変換は、ユーザが使用するのに十分である。他の再生用アプリケーションでは、プレゼンテーションデータは中間形式にすぎず、更なる変換が必要である。例えば、プリンタである表示システム１５２４の場合、プレゼンテーションデータ１５１４を解釈用アプリケーション１５１８によって更に解釈しなくてはならない。解釈用アプリケーション１５１８を、プリンタ内の分析装置とすることができる。解釈用アプリケーション１５１８は、他のシステムリソース１５１６を用いてプレゼンテーションデータ１５１４を画像データ１５２０に変換する。画像データ１５２０の形式は、表示装置１５２２で直接表示する（プリンタの場合、プリント文書として出力する）ことの可能なものである。

【０１６６】再生の際に電子著作物を保護する前述のシステム及び方法に加え、秘話化コンテンツを生成し電子著作物のリソース情報を保存する第１の秘話化方式に従って電子著作物を秘話化することにより、電子著作物を再生の際に保護することができる。電子著作物のリソース情報の一部をコピーし、第２の秘話化方式に従って秘話化する。図１６を参照すると、再生用アプリケーション１６１２は秘話化リソース情報１６１４（及び必要とされうるあらゆる他のシステムリソース情報１６１６）を用いて秘話化電子著作物１６１０を平文のプレゼンテーションデータ１６１８に変換する。プレゼンテーションデータは必然的に平文の形式であり、これは、このプレゼンテーションデータが他のプログラム（スクリーン・キャプチャ・ユーティリティ・プログラムなど）によって入手可能であることを意味する。しかしながら、このような他のプログラムの出力はオリジナルの電子著作物と同一の形式ではなく、同一の忠実度ではないことがよくある。

【０１６７】秘話化リソース情報は、秘話化電子コンテンツを平文の画像（プレゼンテーションデータ）にする秘話化フィルタのように作用するものと考えることができる。あらゆる市販のアプリケーションでありうる再生用アプリケーションが、平文の電子コンテンツを知らない、又は知る必要がないという点で、このシステムは盲目的再生システムである。盲目的再生は任意の変換関数Rに対して作用し、R(w',s')=R(w,s)となる。式中、w'は秘話化電子コンテンツ、wは平文の電子コンテンツ、s'は秘話化リソース情報、及びsは秘話化されていないリソース情報である。秘話化リソース情報を用いた秘話化電子著作物の盲目的再生は、盲目的再生が秘話化解除を必要とせずに平文のプレゼンテーションデータを生成するという点で、前述の盲目的変換とは異なっている。盲目的変換では、再生用アプリケーションは暗号化電子著作物を暗号化プレゼンテーションデータに変換し、それからその復号をしなくてはならない。双方の場合において、ユーザはオリジナルの電子著作物を平文の形式で見ることはない。

【０１６８】秘話化電子著作物及び秘話化リソース情報を用いた盲目的再生（盲目的解釈とも呼ばれる）のみを用いて、通常の暗号化に加え、再生の際に電子著作物を保護することができる。例えば、秘話化電子著作物及び秘話化リソース情報を暗号化して配布の際に電子著作物を保護し、それからユーザシステムでこれらを秘話化電子著作物及び秘話化リソース情報に復号することができる。ユーザはまず、コンテンツ所有者又はコンテンツ所有者に代わって（暗号化電子著作物を復号するために）働く配布業者から許可を得なくてはならない。ユーザが認可されると、暗号化された秘話化電子著作物及び暗号化された秘話化リソース情報を復号し、秘話化リソース情報を用いて秘話化電子著作物を再生用アプリケーションで再生する。

【０１６９】ユーザが表示するための使用可能な形式に電子著作物を解釈することの複雑性を用いて、電子著作物を再生の際に更に保護することができる。図１７を参照すると、秘話化電子著作物１７１０が再生用アプリケーション１７１２に提供されており、再生用アプリケーション１７１２は秘話化システムリソース１７１６及び他のシステムリソース１７１８を用いて秘話化電子著作物１７１０を部分的に秘話化されたプレゼンテーションデータ１７１４に変換する。この実施形態では、プレゼンテーションデータをユーザによって使用可能な形式に変換するには表示システム１７２８が必要である。部分的に秘話化されたプレゼンテーションデータ１７１４は解釈用アプリケーション１７２０に提供され、解釈用アプリケーション１７２０は秘話化システムリソース１７１６、ローカルシステムリソース１７２２、及びシステムリソース１７１８を用いて、部分的に秘話化されたプレゼンテーションデータ１７１４を平文の画像データ１７２４に変換する。次に、ユーザの使用のために平文の画像データ１７２４を表示装置１７２６上に表示する。この実施形態では、プレゼンテーションデータはなお秘話化されており、平文データの場所を表示処理のうちでもより後の時点まで分からないようにし、更なる保護を提供している。

【０１７０】電子著作物を秘話化するシステムの有用性を高めるために、秘話化リソース情報を電子著作物から分離させ、スマートカードなどの持運び可能なデバイスに関連させてもよい。この実施形態では、再生用アプリケーション１７１２は秘話化システムリソース１７１６を用いて著作物を再生する。秘話化システムリソース１７１６をローカルメモリに記憶する代わりに、秘話化システムリソース１７１６を秘話化電子著作物１７１０と共にスマートカードなどの持運び可能なデバイスに記憶する。また、ハードウェア強化機能を有しうるスマートカードは、変更を困難にする属性を備えることができる。持運び可能なコンテクスト内では、秘話化データを再生用アプリケーション１７１２によって処理して部分的に秘話化されたプレゼンテーションデータを生じ、それからそのデータを解釈用アプリケーション１７２０に提供する。

【０１７１】多くの様々なタイプの電子著作物を、その使用にわたり秘話化方式を用いて保護することができる。例えば、電子著作物が文書又はテキストファイルである場合、再生用アプリケーションをワードプロセッサとすることができ、システムリソース又はリソース情報はフォントテーブル、ページレイアウト、及びカラーテーブルを含むことができる。電子著作物がオーディオ又はビデオデータ（例えばストリーム）である場合、再生用アプリケーションをオーディオ又はビデオプレーヤーとすることができる。プレゼンテーションデータはオーディオ／ビデオ最終データのストリームである。表示システムを、オーディオ／ビデオ装置とすることができる。解釈用アプリケーションを、オーディオ／ビデオ装置のドライバとすることができる。画像データをオーディオ／ビデオ装置のデータストリームとし、表示装置をオーディオ／ビデオ装置の解釈装置（例えばスピーカー又はモニタ）とすることができる。

【０１７２】オーディオ／ビデオのデータストリームである電子著作物に関しては、システムリソース又はリソース情報は、オーディオ／ビデオ装置の特徴、即ちサンプリング・レート（１秒あたりのサンプリング、例えば8kHz、44.1kHz）、サンプリング品質（サンプリングあたりのビット数、例えば８、１６）、サンプリング・タイプ（チャネルの数、例えばモノラルは１、ステレオは２）、及びサンプリング形式（命令及びデータのブロック）を含むことができる。秘話化のために選択可能ないくつかのオーディオ／ビデオデータストリーム及び対応するリソース情報又は可変パラメータの表を以下に示す。

【０１７３】

【表１】

拡張子	出所	可変パラメータ（#は固定パラメータ）	圧縮	プレーヤー
.mp3	MPEG 標準	サンプリング・レート、品質、#タイプ	MPEG	MP3 Player
.ra	Real Networks	サンプリング・レート、品質、#タイプ	Plug-ins	Real Player
.wav	Microsoft	サンプリング・レート、品質、#タイプ	ADPCM	Window Media
.snd	Apple	サンプリング・レート、#品質、#タイプ	MACE	QuickTime

表１：電子著作物：A/V データ（ストリーム）

【０１７４】電子著作物の構造を、秘話化のために有意に使用することができる。電子著作物全体を秘話化することができるが、電子著作物の一部のみを秘話化することがより好ましい。電子著作物の殆どは、命令、データ、及びリソースといった３つの主要エレメントを含んでいる。前述の形式保存暗号化方法とほぼ同様に、電子著作物のデータ及びリソースのみを秘話化することが好ましい。データ及びリソースのみを選択的に変換することにより、コンテンツは元の形式のままであるがデータ及びリソースは理解できなくなるように電子著作物を変換することができる。

【０１７５】文書タイプの電子著作物の一般的なレイアウトを図１８に示す。図１８において、電子著作物１５０はページ(Page)記述子１５２、制御(Control)コード１５４、１５８及び１６２、リソース(Resource)識別子１５６、並びにデータ(Data)１６０及び１６４を含む。ページ記述子１５２は、著作物の一般的なレイアウトを定める。例えば、ページサイズ、ページ数、及び余白は、電子文書に関してはページ記述子の範囲に属する。制御コード１５４、１５８及び１６２は、コンテンツのプレゼンテーションを記述するという点で類似している。例としては、テキスト位置設定コマンド、テキスト出力コマンド、フォントタイプ設定コマンド、及び現行画面の座標設定コマンドが挙げられる。リソース識別子１５６は単に所望のリソースを参照する。電子文書の分野では、リソースは書体から背景色まで様々である。最

後に、データ１６０及び１６４は、電子著作物によって伝達される情報の核を表す。これは、マルチメディアクリップに用いられる描画座標、又は電子文書として解釈するための文字コードでありうる。

【０１７６】電子著作物（この場合は簡潔な電子文書）とその秘話化形式の１つの例を、平文及び秘話化された形式のHTML文書として図１９及び図２０に示す。<html>及び<body>のタグはページ記述子である。<font>から<＼font>までのタグはフォントリソース特性を設定する制御コードの一例であり、"Arial"及び"14"はArial書体で１４ポイントのフォントのためのリソース識別子である。"Hello World"のテキストはデータ、即ち著作物の情報の核である。<p>は、段落の初めを示す他の制御コードである。最後に、文書は文書の終わりを識別するためのページ記述子<＼body>及び<＼html>で終わっている。

【０１７７】図２０は、図１９の電子著作物が秘話化形式でどのように見えるかを示している。ページ記述子及び制御コードのタグは変わらぬままであり、即ち、<html>、<body>及び<font>のタグは変わっていないことがわかる。これに対し、リソース識別子、"Arial"及び"14"は復号不可能な値に変換されている。同様に、データである"Hello World"も、復号不可能な値に変換されている。リソース識別子及びデータを変換することにより、コンテンツは秘話化形式であり、意味のないものになる。しかしながら、ページ記述子及び制御コードが不変のままであるという事実により、文書は元の形式を保つことができ、元の形式としては一般にHTML、Adobe社のPDF、RealNetworks社のRAM、Apple社のQuickTimeなどが可能である。

【０１７８】システムコンテクスト（又はシステムリソース又はリソース情報）を、特定のシステムの再生用アプリケーションに対して利用可能なシステムリソースの集まりとして考えることができる。例えば、システムコンテクストは、フォントテーブル、色パレット、システム座標及び音量設定を含みうる。電子著作物を再生用アプリケーションに入力すると、再生用アプリケーションは電子著作物内に含まれる特定のリソース情報を用いて電子コンテンツをプレゼンテーションデータに変換する。電子著作物内に含まれる各システムコンテクスト又はリソース情報をシステムに対して固有になるように変更し、そのシステムのために再生可能にすることができる。システムコンテクストは電子著作物の使用に不可欠なエレメントであり、再生のために、電子著作物の使用を特定のシステム、物理的な装置又は再生用アプリケーションに関連づける。電子著作物内のリソース識別子及びデータは、システムコンテクスト内に含まれるエレメントを直接的又は間接的に参照することができる。電子著作物及びシステムコンテクストの秘話化により、平文のプレゼンテーションデータへの盲目的解釈が可能になる。固有のシステムに関連する秘話化シードを用いてシステムコンテクストを秘話化することにより、生じる秘話化システムコンテクストを固有の環境とすることができ、この固有の環境内で、同一の秘話化シードで秘話化された相補的な秘話化電子著作物にアクセスし、再生することができる。

【０１７９】図２１は、システムコンテクストの一般的な構造を示している。これらのエレメントは、リソース識別子(ResID)、エレメント識別子(ElemID)、及びリソース特性(Characteristics)を含む。ResIDは、リソースを参照するための、他のシステム構成要素の関連情報を含む。ElemIDは、リソース内の個々のエレメントの識別子である。最後に、Characteristicsは、個々のリソースエレメントの表現に用いられる実際のリソース特性である。

【０１８０】図２２は、Arial書体に関するフォントテーブルのリソースの図である。この場合の主要リソース識別子はフォント名"Arial"である。ASCIIの規則にならい、番号４８は個々のリソースエレメント識別子を識別する。ElemIDのリソースエレメント特性は、文字'a'を表すための情報を表している。

【０１８１】図２３は、図２２に示すフォントリソースの秘話化システムコンテクストの図である。リソース識別子そのものが"k13k2"に変換されている。エレメント識別子そのものは、リソース特性のみを変換するには十分であるため、変換の必要はない。この場合、"48"は、'a'の代わりに'Y'の特性を表すために変換されたものとして示されている。

【０１８２】秘話化及び盲目的解釈を、多くの様々なタイプの電子著作物に対して使用することができる。秘話化及び盲目的解釈を、文書の他にオーディオ／ビデオデータに対して用いることができる。前述のように、オーディオ／ビデオデータは一般にストリームの形式で提供される。再生用アプリケーションは、デジタルオーディオ／ビデオストリームを最終データストリームに変換するオーディオ／ビデオプレーヤーであり、この最終データストリームを変換装置（スピーカー）によってオーディオ出力に処理したり、表示によってビデオ画像に処理したりすることができる。

【０１８３】図１７を参照すると、再生用アプリケーション１７１２はオーディオ／ビデオプレーヤーに相当し、これは一般に、目的のオーディオ／ビデオ装置によって許容されたあるサンプリング・レート、品質及びタイプでオーディオ／ビデオ入力ストリーム１７１０を抽出することによって作動する。オーディオ／ビデオプレーヤーは、オーディオ／ビデオストリームの抽出、混合及び生成にオーディオ／ビデオシステムリソースを用い、次に再抽出されたオーディオ／ビデオストリームを混合して、最終的なオーディオ／ビデオストリームを目的装置が望む形式で生成する。オーディオ／ビデオプレ

ーヤーの場合、プレゼンテーションデータ１７１４は、目的のオーディオ／ビデオ装置が望むサンプリング・レート、品質、タイプ、及び形式である、最終の混合オーディオ／ビデオストリームである。

【０１８４】目的のオーディオ／ビデオ装置（例えば、解釈用アプリケーション１７２０）は、オーディオ／ビデオストリーム（プレゼンテーションデータ１７１４）を特定のサンプリング・レート、品質、タイプ（チャネル）、及び形式（例えば、PAL又はNTSC）で装置のオーディオ／ビデオデータ１７２４に変換することのできる、ハードウェアシステムである。オーディオ装置の例としては、サウンドカード、スピーカー、モニタ、及びオーディオ／ビデオ装置内に配置されるデジタルーアナログ変換器が挙げられる。装置の多くは、様々なサンプリング・レートの範囲でオーディオ／ビデオストリームを再生することができる。画像データ１７２４（例えば、オーディオ信号又はビデオ画像のストリーム）はオーディオ／ビデオ装置のドライバ１７２０によって生成され、表示装置１７２６によって「消費」される。

【０１８５】例えば、オーディオ／ビデオデータストリームを秘話化するために、このストリームを２つ以上の別個のストリームに分割することができる。一方のストリームは秘話化し、もう一方のストリームは秘話化しない。各ストリームは、関連するサンプリング・レート、チャネル、品質及び形式といった、様々な装置特性（リソース情報）を有することができる。装置特性（ストリームのサンプリング・レート、チャネル、品質及び／又は形式のうち１つ以上）を秘話化して、秘話化リソース情報を生成することもできる。

【０１８６】秘話化オーディオ／ビデオストリームの盲目的再生は、秘話化電子文書と同様にして達成される。再生用アプリケーション（オーディオ／ビデオプレーヤー）は秘話化されていないストリームと秘話化ストリームとを共に混合し、秘話化リソース情報を用いて、目的のオーディオ／ビデオ装置用に、リソース情報の正しいセットを有する秘話化された最終データストリームを生成する。目的装置（１７２０）は、秘話化リソース情報を用いて秘話化データストリームを再生し、平文の音声／視覚効果（１７２４）を生成する。

【０１８７】本発明の例示的な実施形態のいくつかを先に詳しく説明したが、本発明の他の形式、代替物、変更物、及び変形物も同様に作用し、これらが当業者にとって明白であることを認識すべきである。この開示内容は本発明を特定の実施形態に限定する意図ではなく、このような形式、代替物、変更物、及び変形物を全て含むものと意図される。例えば、ソフトウェアの構成要素として説明された前述の本発明の一部をハードウェアとして実施することができる。更に、いくつかの機能ブロックを、本明細書では別個で互いから独立したものとして説明したが、これらの機能ブロックを統合し、単一の汎用コンピュータで実行することもできるし、あるいは、当該分野において認識されるように、更に副機能に分割することもできる。従って、本発明の真の範囲は全ての代替物、変更物及び等価物を含むように意図されており、この範囲は後述の請求の範囲を参照して決定されるべきである。

【図面の簡単な説明】

【図１】安全な環境または安全でない環境における電子文書の作成及び商用配布のモデルを表す最上位レベルブロック図である。

【図２】従来技術による保護電子文書の復号を表すフローチャート図である。

【図３】本発明の簡単な実施形態による保護電子文書の復号を表すフローチャート図である。

【図４】本発明の好適な実施形態による保護電子文書の復号を表すフローチャート図である。

【図５】本発明の１実施形態による自己保護文書のデータ構造を表す機能ブロック図である。

【図６】本発明の１実施形態による自己保護文書の作成及びカスタマイズを表すフローチャート図である。

【図７】ユーザの立場から見た、本実施形態による自己保護文書の処理及び使用時に実行される処置を表すフローチャート図である。

【図８】未解釈・暗号化文書と解釈済み・復号化プレゼンテーションデータ間の考えられるパスを表すグラフである。

【図９】文書形式情報を解釈用に平文の状態にした、本発明による秘話化処理を表すフローチャート図である。

【図１０】本発明による、形式保存暗号化及び高信頼性解釈の方法のブロック図である。

【図１１】トークン化される文書の簡単な例を示す図である。

【図１２】図１１の文書用のトークン辞書を示す図である。

【図１３】図１１の文書用の位置テーブルを示す図である。

【図１４】本発明による秘話化電子著作物及び秘話化システムリソースの生成方法を表すブロック図である。

【図１５】従来技術による電子著作物の画像データへの変換を表すブロック図である。

【図１６】本発明による秘話化電子著作物の盲目的再生システムを表すブロック図である。

【図１７】本発明による秘話化電子著作物の別の盲目的再生システムを表すブロック図である。

【図１８】電子文書の構造の一例を表すブロック図である。

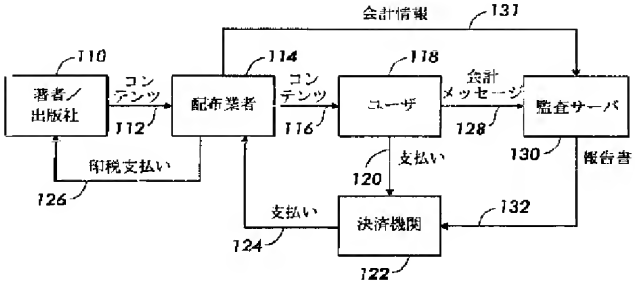
【図１９】電子文書の一例を表す図である。

【図２０】図１６の電子文書を秘話化した後の一例を表す図である。

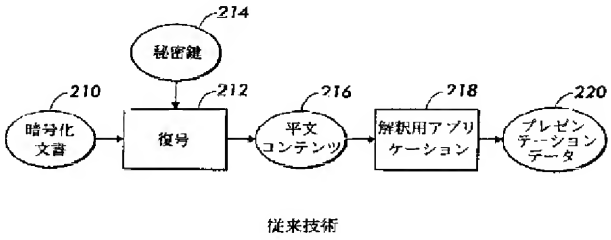
【図２１】電子文書用のリソース情報又はシステムコン

テクストの構造の一例を表すブロック図である。
【図２２】フォントテーブルの一例を表すブロック図である。
【図２３】図２２のフォントテーブルを秘話化した後のブロック図である。
【符号の説明】
１０１０、１４１０、１７１０　電子著作物
１０１２、１６１２、１７１２　再生用アプリケーション
１０１６　暗号化プレゼンテーションデータ
１０１８　復号エンジン
１０２０　プレゼンテーションデータ
１０２２、１７２０　解釈用アプリケーション
１０２４、１７２４　画像データ
１０２６、１７２６　表示装置
１４１２　リソース抽出
１４１４、１６１６、１７１８　システムリソース
１４１６　リソース秘話化
１４１８　秘話化シード
１４２０　コンテンツ秘話化
１４２２、１６１０　秘話化電子著作物
１４２４、１６１４、１７１６　秘話化システムリソース
１４２６　シード生成装置
１６１８　平文のプレゼンテーションデータ
１７１４　部分的に秘話化されたプレゼンテーションデータ
１７２２　ローカルシステムリソース
１７２８　表示システム

【図１】



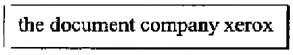
【図２】

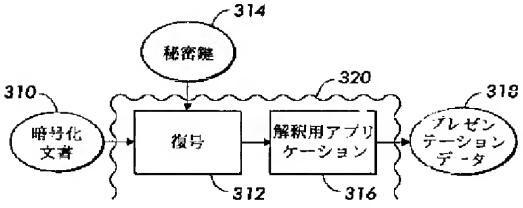


【図３】

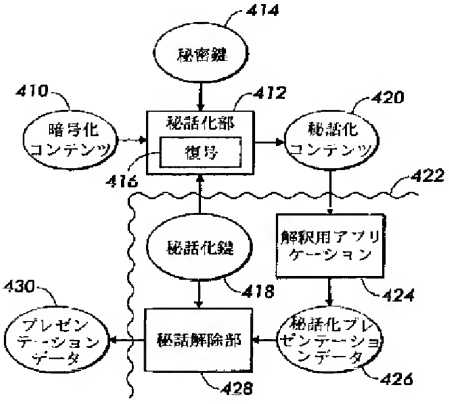
【図４】

【図１１】

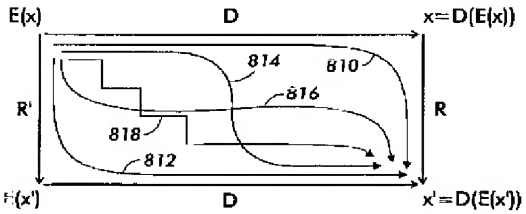




【図８】



【図１２】

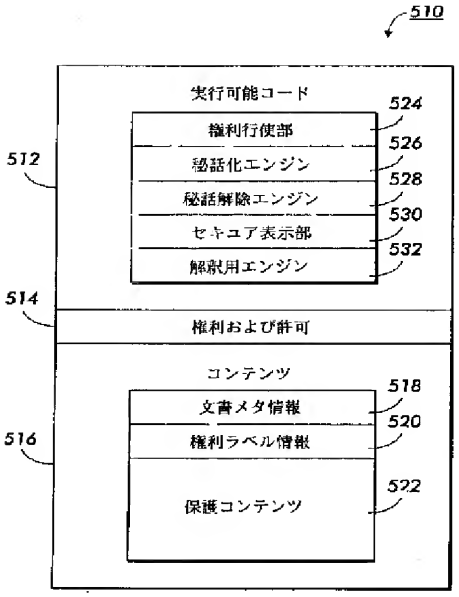


【図１８】

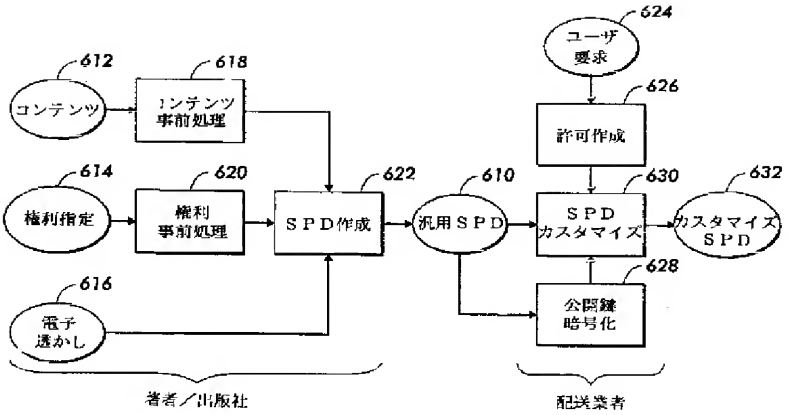
識別子	1000	1001	101	0100	000	001	0101	011	1101	11000	11001	11100	1111	11101
トークン	t	h	e	d	o	c	u	m	n	p	a	y	x	r

Page	Control	Resource	Control	Data	Control	Data	. . .

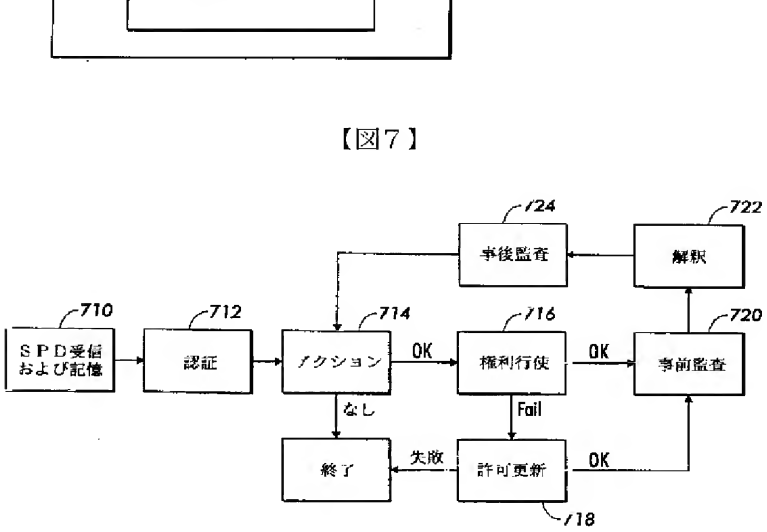
【図５】



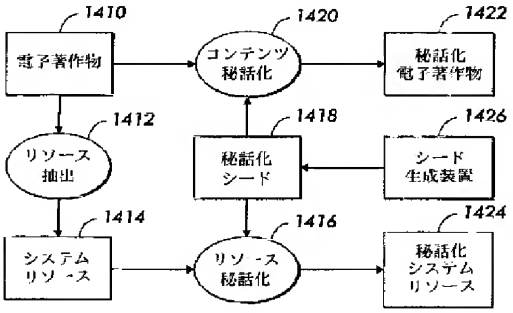
【図６】



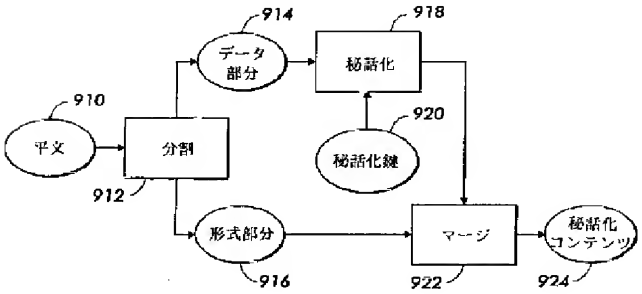
【図７】



【図１４】



【図９】



【図１３】

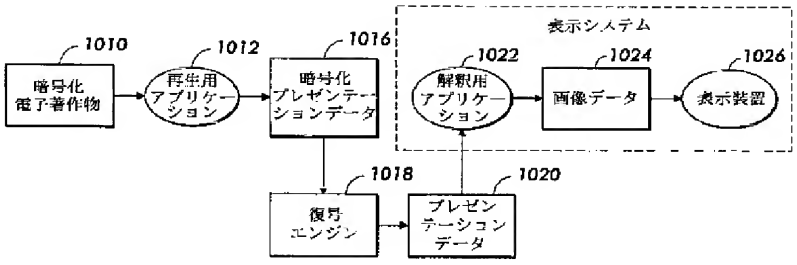
識別子	1000	1001	101	0100	000	001	0101	011	101	1101	1000	001	000	011
x	10	10	10	20	10	10	10	10	10	10	10	20	10	10
y	10	0	0	0	0	0	0	0	0	0	0	0	0	0

識別子	11000	11001	1101	11100	1111	101	11101	000	1111
x	10	10	10	10	20	10	10	10	10
y	0	0	0	0	0	0	0	0	0

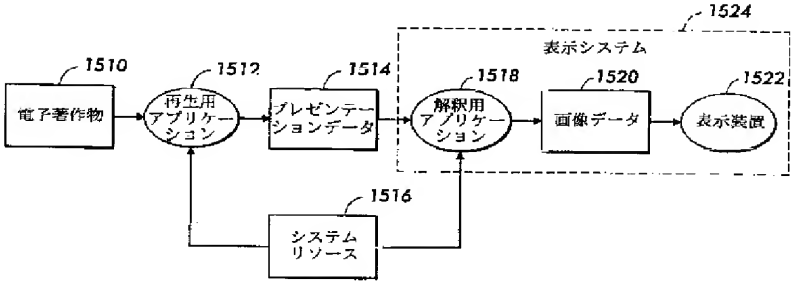
【図１９】

```
<html>
<body>
<font name="Arial" size="14">
Hello World
</font>
<p>
</body>
</html>
```

【図１０】

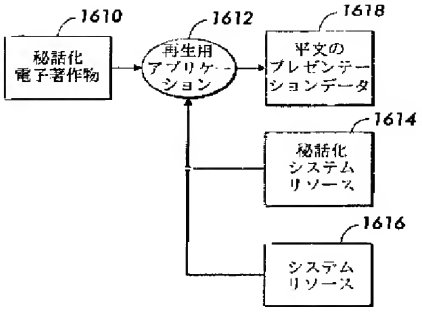


【図１５】



従来技術

【図１６】

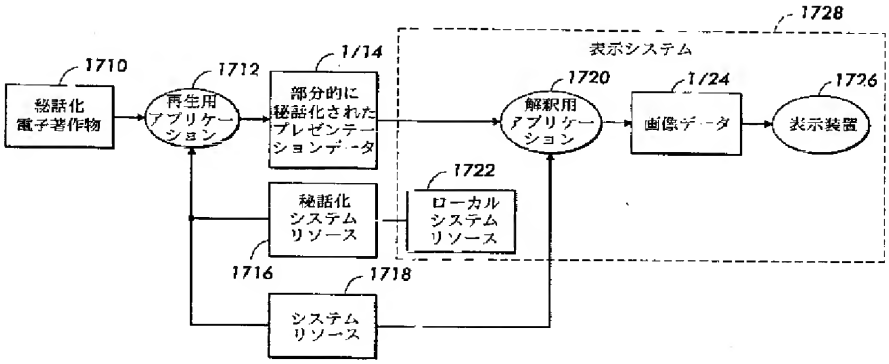


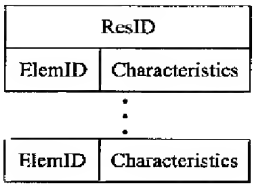
【図２０】

```
<html>
<body>
<font name="kl3k2" size="21">
v0aa 8 aa0
</font>
<p>
</body>
</html>
```

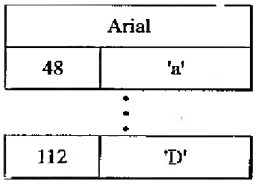
【図２１】

【図１７】

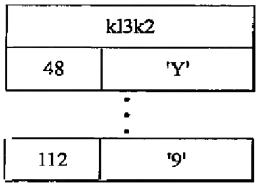




【図２２】



【図２３】



フロントページの続き

(71)出願人　500470703
103　Ｆｏｕｌｋ　Ｒｏａｄ，　Ｓｕｉｔｅ　205－Ｍ，　Ｗｉｌｍｉｎｇｔｏｎ，　Ｄｅｌａｗａｒｅ　19803，　Ｕｎｉｔｅｄ　Ｓｔａｔｅｓ　ｏｆ　Ａｍｅｒｉｃａ

(72)発明者　シン　ワン
アメリカ合衆国　90007　カリフォルニア州　ロサンゼルス　シュライン　プレイス　3005　ナンバー８

Ｆターム(参考)　5B017 AA03 BA07 CA16
5J104 AA16 AA41 EA06 JA19 NA02 PA07

【外国語明細書】

SYSTEM AND METHOD FOR PROTECTION OF DIGITAL WORKS

Inventor: Xin Wang

Copyright Notice

Related Application

This application is a continuation-in-part application of application no. 09/178,529 filed October 23, 1998.

Field of the Invention

This invention relates to document rights management, and more particularly, to a method for protecting digital works which employs format-preserving encryption that enables blind transformations.

Background of the Invention

One of the most important issues impeding the widespread distribution of digital documents or works via electronic commerce is the current lack of protection of the intellectual property rights of content owners during the distribution and use of those digital documents or works. Efforts to resolve this problem have been termed "Intellectual Property Rights Management" ("IPRM"), "Digital Property Rights Management" ("DPRM"), "Intellectual Property Management" ("IPM"), "Rights Management" ("RM"), "Digital Rights Management" ("DRM") and "Electronic Copyright Management" ("ECM"). At the core of Digital Rights Management is the underlying issue of ensuring that only authorized users may perform operations on digital

documents or works that they have acquired. Once accessed, the content must not be distributed or used in violation of the content owner's specification of rights.

A document or work, as the term is used herein, is any unit of information subject to distribution or transfer, including but not limited to correspondence, books, magazines, journals, newspapers, other papers, software, photographs and other images, audio and video clips, and other multimedia presentations. A document may be embodied in printed form on paper, as digital data on a storage medium, or in any other known manner on a variety of media. A digital work, as the term is used herein, is any document, text, audio, multimedia or other type of work or portion thereof maintained in a digital form that can be replayed or rendered using a device or a software program.

In the world of printed documents, a work created by an author is usually provided to a publisher, which formats and prints numerous copies of the work. The copies are then sent by a distributor to bookstores or other retail outlets, from which the copies are purchased by end users.

While the low quality of copying and the high cost of distributing printed material have served as deterrents to the illegally copying of most printed documents, it is far too easy to copy, modify, and redistribute unprotected electronic documents. Accordingly, some method of protecting electronic documents is necessary to make it harder to illegally copy them. This will serve as a deterrent to copying, even if it is still possible, for example, to make hardcopies of printed documents and duplicate them the old-fashioned way.

With printed documents, there is an additional step of digitizing the document before it can be redistributed electronically; this serves as a deterrent. Unfortunately, it has been widely recognized that there is no viable way to prevent people from making unauthorized distributions of electronic documents within current general-purpose computing and communications systems such as personal computers, workstations, and other devices connected over local area networks (LANs), intranets, and the Internet. Many attempts to provide hardware-based solutions to prevent unauthorized copying have proven to be unsuccessful.

Two basic schemes have been employed to attempt to solve the document protection problem: secure containers (systems which rely on cryptographic mechanisms) and trusted systems.

Cryptographic mechanisms encrypt (or "encipher") documents that are then distributed and stored publicly, and ultimately privately decrypted by authorized users. Cryptographic mechanisms provide a basic form of protection during document delivery from a document distributor to an intended user over a public network, as well as during document storage on an insecure medium. Many digital rights management solutions rely on encrypting the digital work and distributing both the encrypted message and decryption key to the consumer's system. While different schemes are employed to hide the decryption key from the consumer, the fact remains that all necessary information is available for a malicious user to defeat the protection of the digital work. Considering that current general-purpose computers and consumer operating systems provide little in the way of sophisticated security mechanisms, the threat is both real and obvious.

A "secure container" (or simply an encrypted document) offers a way to keep document contents encrypted until a set of authorization conditions are met and some copyright terms are honored (e.g., payment for use). After the various conditions and terms are verified with the document provider, the document is released to the user in clear form. Commercial products such as IBM's Cryptolopes and InterTrust's Digiboxes fall into this category. Clearly, the secure container approach provides a solution to protecting the document during delivery over insecure channels, but does not provide any mechanism to prevent legitimate users from obtaining the clear document and then using and redistributing it in violation of content owners' intellectual property.

Cryptographic mechanisms and secure containers focus on protecting the digital work as it is being transferred to the authorized user/purchaser. However, a digital work must be protected throughout its use from malicious users and malicious software programs. Even if a user is a trusted individual, the user's system may be susceptible to attack. A significant problem facing electronic commerce for digital works is ensuring that the work is protected on the target consumer's device. If the protection for the digital

work is compromised, valuable and sensitive information is lost. To complicate matters, today's general-purpose computers and consumer operating systems are deficient in the areas of security and integrity. Protecting the work throughout usage is a much more complex issue that remains largely unsolved.

In the "trusted system" approach, the entire system is responsible for preventing unauthorized use and distribution of the document. Building a trusted system usually entails introducing new hardware such as a secure processor, secure storage and secure rendering devices. This also requires that all software applications that run on trusted systems be certified to be trusted. While building tamper proof trusted systems is still a real challenge to existing technologies, current market trends suggest that open and untrusted systems such as PC's and workstations will be the dominant systems used to access copyrighted documents. In this sense, existing computing environments such as PC' s and workstations equipped with popular operating systems (e.g., Windows and UNIX) and render applications (e.g., Microsoft Word) are not trusted systems and cannot be made trusted without significantly altering their architectures.

Accordingly, although certain trusted components can be deployed, users must continue to rely upon various unknown and untrusted elements and systems. On such systems, even if they are expected to be secure, unanticipated bugs and weaknesses are frequently found and exploited.

Conventional symmetric and asymmetric encryption methods treat messages to be encrypted as basically binary strings. Applying conventional encryption methods to documents has some drawbacks. Documents are typically relatively long messages; encrypting long messages can have a significant impact on the performance of any application that needs to decrypt the document prior to use. More importantly, documents are formatted messages that rely on appropriate rendering applications to display, play, print and even edit them. Since encrypting a document generally destroys formatting information, most rendering applications require the document be decrypted into clear form before rendering it. Decryption prior to rendering opens the possibility of

disclosing the document in the clear after the decryption step to anyone who wants to intercept it.

There are a number of issues in rights management: authentication, authorization, accounting, payment and financial clearing, rights specification, rights verification, rights enforcement, and document protection. Document protection is a particularly important issue. After a user has honored the rights of the content owner and has been permitted to perform a particular operation with a document (e.g., print it, view it on-screen, play the music, or execute the software), the document is presumably in-the-clear, or unencrypted. Simply stated, the document protection problem is to prevent the content owner's rights from being compromised when the document is in its most vulnerable state: stored, in the clear, on a machine within the user's control.

Even when a document is securely delivered (typically in encrypted form) from a distributor to the user, it must be rendered to a presentation data form before the user can view or otherwise manipulate the document. Accordingly, to achieve the highest level of protection, it is important to protect the document contents as much as possible, while revealing them to the user at a late stage and in a form that is difficult to recover into a useful form.

In the known approaches to electronic document distribution that employ encryption, an encrypted document is rendered in several separate steps. First, the encrypted document is received by the user. Second, the user employs his private key (in a public key cryptosystem) to decrypt the data and derive the document's clear content. Finally, the clear content is then passed on to a rendering application, which translates the computer-readable document into the finished document, either for viewing on the user's computer screen or for printing a hardcopy. The clear content is required for rendering because, in most cases, the rendering application is a third-party product (such as Microsoft Word or Adobe Acrobat Reader) that requires the input document to be in a specific format. It should be appreciated, then, that between the second and third steps, the previously protected document is vulnerable. It has been decrypted, but is still stored in clear electronic form on the user's computer. If the user is careless or is otherwise

motivated to minimize fees, the document may be easily redistributed without acquiring the necessary permissions from the content owner.

While no system is completely spoof proof or immune to attack, some recent techniques protect digital works by limiting use of the digital work to a user-specified physical device. These techniques require the user to provide private information or system state information from the system or physical device the user intends to use to render the digital work. System state information is typically defined as system configuration information such as system parameters, CPU identifier, device identifiers, NIC identifiers, drive configuration, etc. In these techniques, the digital content is encrypted using a session key, then the session key, rather than using the user's encryption key, is encrypted using a combination of the system or state information and the user's credentials. Then both the encrypted content and key are transmitted to the destination repository. In order to use the received encrypted work, the user must contact a trusted authorizing entity (usually a remotely located software program) which verifies the user's identity and credentials, then together with system state, decrypts the session key and finally decrypts the content for use.

Commercial applications such as the secure Adobe Acrobat reader and the secure Microsoft MediaPlayer validate usage of the digital work by checking a license voucher for the appropriate user credentials and usage rights. Among the user credentials are system device identifiers such as the CPU identifier or certain device serial numbers. At the time the user invokes an operation on the digital work, the application verifies if the specified device is present. This provides assurance that the digital work has not been transmitted to an unauthorized user (actually to an unauthorized device). While the programmatic check provides a minimal level of assurance, it depends on the security of the secret, which resides on the user's device. Not only can the decryption key be violated, but also the device identifiers themselves are particularly susceptible to the threat of spoofing.

The Acrobat Reader and MediaPlayer protection schemes operate by allowing the rendering application to identify required devices on the user system as specified in the

license voucher issued for the digital work. This provides a level of protection adequate in many circumstances (i.e., if the user is trusted and the user's specified rendering device is not susceptible to attack). The weakness of the schemes is that it is based on the assumption that neither the protection of the cryptographic key nor the integrity of the license voucher will be compromised.

These techniques are really more of an authentication technique than a protection technique, in that once the user's identity and credential information, system state information is verified or license voucher received, the content is decrypted to its clear state and then becomes vulnerable to attack. The digital work is afforded no protection throughout usage. Further, the user information approach is problematic in that it assumes the user will be sufficiently deterred from passing along his/her personal information. In other words, for the user information approach to succeed there must be severe consequences for users who would reveal their private identity and credential information.

A significant drawback to the schemes which tie authorization a specific device is that they require the user to divulge sensitive information (e.g., CPU number or other personal information) which raises a concern regarding privacy issues. While the user divulges the information voluntarily (the user's only option if he/she does not wish to divulge this information is not to receive the digital work) it would be desirable to provide a protection scheme that could secure a digital work on a user's device without requiring private information. It would also be desirable to provide a DRM solution which does not rely on the protection of the cryptographic key or the integrity of the license voucher. It would be desirable to provide a DRM solution which delayed decryption of the digital content to the latest possible moment.

Accordingly, it would be beneficial to provide an electronic document distribution scheme that minimizes the disadvantages of known systems. Such a scheme would prevent users from obtaining a useful form of an electronically-distributed document during the decryption and rendering processes.

Summary of the Invention

A self-protecting document ("SPD"), according to the invention, is not subject to the above-stated disadvantages of the prior art. By combining an encrypted document with a set of permissions and an executable code segment that includes most of the software necessary to extract and use the encrypted document, the self-protecting document accomplishes protection of document contents without the need for additional hardware and software.

The SPD system is broken down between a content creator (analogous to the author and the publisher of the traditional model) and a content distributor. The author/publisher creates the original document, and decides what rights are to be permitted. The distributor then customizes the document for use by various users, ensuring via the customization that the users do not exceed the permissions they purchased.

At the user's system, the self-protecting document is decrypted at the last possible moment. In an embodiment of the invention, various rendering facilities are also provided within the SPD, so that the use of the SPD need not rely upon external application that might not be trustworthy (and that might invite unauthorized use). In an alternative embodiment, interfaces and protocols are specified for a third-party rendering application to interact with the SPD to provide trusted rendering.

In one embodiment of the invention, the encrypted document is decrypted by the user's system while simultaneously "polarizing" it with a key that is dependent, at least in part, on the state of the user's system. The polarization may be cryptographically less secure than the encryption used for distribution, but serves to deter casual copying. In this embodiment, depolarization is performed during or after the rendering process, so as to cause any intermediate form of the document to be essentially unusable.

In another embodiment of the invention, a method of protecting a digital work uses a blind transformation function to transform an encrypted digital work into encrypted presentation data. The originator's digital content is protected in its original form by not being decrypted. This method enables the rendering or replay application to process the

encrypted document into encrypted presentation data without decrypting it first. Encrypted presentation data is then decrypted just before it is displayed to the user. This method improves the overall performance of the process (both decryption and rendering) by minimizing the decryption overhead (since pre-rendering decryption is generally more time and resource consuming) and postponing the decryption to a late stage of the rendering process.

Blind transformation or blind computing can be accomplished in one of several ways. Most digital works include formatting information, which when encrypted cannot be processed by the replay or rendering application (the transformation function which transforms a digital work into presentation data). If the digital work is encrypted with a format preserving encryption scheme, any transformation function may be used. This is particularly useful in that any commercial replay or rendering application can process the encrypted digital work into encrypted presentation data. Otherwise, the blind transformation function is a function of the original transformation function. For example, the blind transformation function may be a polynomial of the original transformation function. Alternatively, both the blind transformation function and the original transformation function may be any multivariate, integer coefficient affine function.

Not all encryption schemes are format preserving encryption schemes. Additive encryption schemes may be used with all document types and all associated transformation functions. In some replay or render applications, for some types of documents, portions of the format information may be left in the clear. In other types of documents all of the format information may be encrypted. In some types of documents, an additive encryption scheme may be used to encrypt the format information and any encryption scheme may be used to encrypt the content or data portion of the document.

In particular, additive encryption schemes can be used to encrypt coordinate information of documents so that some rendering transformations can be performed on the encrypted coordinate data. In a special class of documents, token-based documents, for example, there are two places during the format-preserving encryption that use

encryption schemes: one is for coordinate or location information x and y of the particular tokens within the document, and the other is for the dictionary of individual token images. In order to perform blind transformation on the individual coordinates of the particular tokens in the document, the first encryption scheme must be an additive encryption scheme. However, the token dictionary may be encrypted with any encryption scheme.

An encrypted token dictionary may still leak information such as the sizes of the token images. If this is a concern (such as if the token dictionary is small), the tokens can be padded with some extra bits before encryption. The padding can result in encrypted token images of a same size or several fixed sizes. For a token-based document, the coordinate information of the tokens in the dictionary may not be encoded. If it is desired that coordinate information be encoded, say, as Huffman codewords, the same approach that is used to encrypt the identifiers can be used to deal with this situation. Basically, the codewords in location tables are left in the clear, and the codewords in the codeword dictionary are hashed using some one-way hash function and their corresponding coordinate information is encrypted. During rendering the codewords in the location tables are first hashed and then used to lookup their encrypted coordinate information.

In another embodiment of the invention, a digital work and a system context (or resource information or system resource) are polarized enabling trusted rendering or replay of the digital work without depolarization of the digital content. In this embodiment, the digital work is of the type which includes digital content and resource information. Resource information may include information used by a replay application to format or process the digital work into presentation data. Resource information may include, for example, a collection of system resources available to the replay software on a particular system, such as the Font Table, Color Palette, System Coordinates and Volume Setting.

Different types of digital works may be polarized. In addition to polarizing typical document type digital works, audio and video digital works can be polarized. The digital work and system context are usually polarized at a manufacturer or content

owner's location using a polarization engine. A polarization engine is a component used to transform the digital work and system context to their respective polarized forms. The polarization engine employs a polarization scheme which relies on some polarization seed, an element used to initialize and customize the polarization engine.

Various polarization schemes may be used to polarize a digital work. For example, a stateless polarization employs a random number as a seed to transform a digital work into a polarized digital work. A state-based polarization scheme employs a seed based on a system state or characteristic of a system to transform a digital work into a polarized digital work that is associated with that system state or characteristic. A dynamic state based polarization scheme employs a seed based on a dynamic system state or characteristic to transform a digital work into a polarized digital work. In this embodiment, the polarized digital work will typically be provided with a polarization engine for repolarizing the encoded digital work and the encoded system context according to the dynamic state-based polarization scheme each time the system requests replay of the digital work. An authorization-based polarization scheme employs a seed based on authorization information received from a trusted source to transform a digital work into a polarized digital work. For further security, the polarized system context can be stored separately from the polarized digital work in a removable context device, which must be coupled to the system prior to use of the digital work.

Preferably the polarization seed contains information which can be used to tie the particular digital work to the ultimate end user or an ultimate end user system. Typically the owner or distributor will select the type of polarization scheme to be used in polarizing the digital work and the type of polarization key to use depending on the value of the digital work. Like encryption schemes, polarization schemes come in different levels of complexity and strength. When a digital work is ordered, a copy of a portion of the digital work's resource information, called the system context, is made. The polarization seed is selected and both the digital work and the system context are polarized. A different polarization scheme may be used for the system context than is used for the digital work. However the polarization seed is the same for both. The

polarized digital work and polarized system context are then provided to the user for replay or rendering on a replay or rendering system.

In the format preserving encryption and trusted rendering embodiment of the invention, protection is provided until the encrypted presentation data must be decrypted into clear presentation data. In this embodiment of the invention, the replay application uses the polarized resource information to transform a polarized digital work into clear presentation data.

If only the digital content of a digital work is polarized, leaving the resource information unpolarized or in the clear, the replay application will be able to process the polarized digital work into polarized presentation data. This means a depolarizer must depolarize the presentation data into clear presentation data suitable for viewing or use by the user. If a portion of a digital work's resource information is also polarized accordingly, when the replay application transforms the polarized digital work, the replay application uses the polarized system resource information to transform the polarized digital work into clear presentation data. All or just a portion of the required resource information may be polarized. The replay is blind in that the replay application does not see the original, unpolarized digital content.

In this embodiment, a polarized digital work is transformed by the replay application using a polarized system context (resource information) to create clear presentation data; the replay application can be any commercial or third party application. The replay application need not be customized to depolarize the presentation data and no depolarizer engine is required. The replay application operates as a blind replay system (it processes polarized digital content using polarized system resources) and relies on a type of polarization which transforms or encodes the digital work such that the ability to replay it using a software program or device is tied to a specific resource information, thus protecting the content throughout use.

Unlike systems which employ encryption to protect the digital work and eventually decrypt the digital work into its clear form before the digital work is provided to the replay application, the blind replay system keeps the digital work encoded in the

polarized form (there is no explicit decoding step in the blind reply) until the last possible moment of the replay process. In the blind replay system, the polarized digital work itself is never depolarized in the clear. Since presentation data is generally of a lesser quality than the original digital work, even if the presentation data is captured in its clear form, it cannot be easily (if at all) transformed back into the original digital work.

Many different types of digital works and their resource information may be polarized and replayed in a blind replay system. Digital works such as documents, text, audio files, graphics files and video files may be replayed in the blind replay system of the invention by polarization of an appropriate resource information.

Brief Description of the Drawings

The structure and function of the invention is best understood with reference to the included drawings, which may be described as follows:

FIGURE 1 is a top-level block diagram representing a model for the creation and commercial distribution of electronic documents in either secure or insecure environments;

FIGURE 2 is a flow diagram illustrating the decryption of protected electronic documents according to the art;

FIGURE 3 is a flow diagram illustrating the decryption of protected electronic documents according to a simple embodiment of the invention;

FIGURE 4 is a flow diagram illustrating the decryption of protected electronic documents according to a preferred embodiment of the invention;

FIGURE 5 is a functional block diagram illustrating the data structures present in a self-protecting document according to an embodiment of the invention;

FIGURE 6 is a flow diagram illustrating the creation and customization of a self-protecting document according to an embodiment of the invention;

FIGURE 7 is a flow diagram, from a user's perspective, illustrating the actions performed in handling and using a self-protecting document according to the invention;

FIGURE 8 is a graph illustrating several possible paths between an unrendered and encrypted document, and rendered and decrypted presentation data;

FIGURE 9 is a flow diagram illustrating a polarization process according to the invention in which document format information remains in the clear for rendering.

FIGURE 10 is a block diagram of a method of format preserving encryption and trusted rendering according to the invention;

FIGURE 11 is a simple example of a document to be tokenized;

FIGURE 12 is the token dictionary for the document of Fig. 11;

FIGURE 13 is the location table for the document of Fig. 11;

FIGURE 14 is a block diagram illustrating a process for generating a polarized digital work and polarized system resource according to the invention;

FIGURE 15 is a block diagram illustrating the conversion of a digital work into image data according to the art;

FIGURE 16 is a block diagram illustrating a system for blind replay of a polarized digital work according to the invention;

FIGURE 17 is a block diagram illustrating another system of blind replay of a polarized digital work according to the invention;

FIGURE 18 is a block diagram of an example structure of a digital document;

FIGURE 19 is an example digital document;

FIGURE 20 is an example of the digital document of Fig. 16 after it has been polarized;

FIGURE 21 is block diagram of an example structure of a resource information or system context for a digital document;

FIGURE 22 is a block diagram of an example font table; and

FIGURE 23 is block diagram of the font table of Fig. 22 after it has been polarized.

Detailed Description of the Preferred Embodiments

The invention is described below, with reference to detailed illustrative embodiments. It will be apparent that the invention can be embodied in a wide variety of forms, some of which may be quite different from those of the disclosed embodiments.

Consequently, the specific structural and functional details disclosed herein are merely representative and do not limit the scope of the invention.

Figure 1 represents a top-level functional model for a system for the electronic distribution of documents, which as defined above, may include correspondence, books, magazines, journals, newspapers, other papers, software, audio and video clips, and other multimedia presentations.

An author (or publisher) 110 creates a document's original content 112 and passes it to a distributor 114 for distribution. Although it is contemplated that the author may also distribute documents directly, without involving another party as a distributor, the division of labor set forth in Figure 1 is more efficient, as it allows the author/publisher 110 to concentrate on content creation, and not the mechanical and mundane functions taken over by the distributor 114. Moreover, such a breakdown would allow the distributor 114 to realize economies of scale by associating with a number of authors and publishers (including the illustrated author/publisher 110).

The distributor 114 then passes modified content 116 to a user 118. In a typical electronic distribution model, the modified content 116 represents an encrypted version of the original content 112; the distributor 114 encrypts the original content 112 with the user 118's public key, and modified content 116 is customized solely for the single user 118. The user 118 is then able to use his private key to decrypt the modified content 116 and view the original content 112.

A payment 120 for the content 112 is passed from the user 118 to the distributor 114 by way of a clearinghouse 122. The clearinghouse 122 collects requests from the user 118 and from other users who wish to view a particular document. The clearinghouse 122 also collects payment information, such as debit transactions, credit card transactions, or other known electronic payment schemes, and forwards the collected users' payments as a payment batch 124 to the distributor 114. Of course, it is expected that the clearinghouse 122 will retain a share of the user's payment 120. In turn, the distributor 114 retains a portion of the payment batch 124 and forwards a payment 126 (including royalties) to the author and publisher 110. In one embodiment of this scheme,

the distributor 114 awaits a bundle of user requests for a single document before sending anything out. When this is done, a single document with modified content 116 can be generated for decryption by all of the requesting users. This technique is well-known in the art.

In the meantime, each time the user 118 requests (or uses) a document, an accounting message 128 is sent to an audit server 130. The audit server 130 ensures that each request by the user 118 matches with a document sent by the distributor 114; accounting information 131 is received by the audit server 130 directly from the distributor 114. Any inconsistencies are transmitted via a report 132 to the clearinghouse 122, which can then adjust the payment batches 124 made to the distributor 114. This accounting scheme is present to reduce the possibility of fraud in this electronic document distribution model, as well as to handle any time-dependent usage permissions that may result in charges that vary, depending on the duration or other extent of use.

The foregoing model for electronic commerce in documents, shown in Figure 1, is in common use today. As will be shown in detail below, it is equally applicable to the system and method set forth herein for the distribution of self-protecting documents.

Turning now to Figure 2, the steps performed by the user 118 (Figure 1) in a prior art system for electronic document distribution are shown. As discussed above, cryptographic mechanisms are typically used to encipher documents. Those encrypted documents are then distributed and stored publicly and deciphered privately by authorized users. This provides a basic form of protection during document delivery from a document distributor to an intended user over a public network, as well as during document storage on an insecure medium.

At the outset, an encrypted document 210 is received by the user 118 and passed to a decryption step 212. As is well known in the art, the decryption step 212 receives the user 118's private key, which is stored locally at the user's computer or entered by the user when needed. The document 210 is decrypted, resulting in clear content 216 similar or identical to the original content 112 (Figure 1).

The clear content 216 is passed to a rendering application 218, which constructs presentation data 220, or a usable version of the document's original content 112. In typical systems of this kind, the presentation data 220 is data immediately suitable for display on a video screen, for printing as a hardcopy, or for other use depending on the document type.

As discussed above, the document is vulnerable in systems like this. The clear content 216 can be copied, stored, or passed along to other users without the knowledge or consent of the distributor 114 or the author/publisher 110. Even a legitimate user may be tempted to minimize the licensing fees by capturing the document in the clear in order to redistribute and use it at will, without honoring the intellectual property of the content owners. As discussed above, the present invention is directed to a scheme for preventing such a user from obtaining a useful form of the document during the rendering process on the user's system.

Accordingly, the system and method of the present invention sets forth an alternative scheme for handling encrypted documents at the user 118's system. A simple embodiment of this scheme is illustrated in Figure 3.

Figure 3 looks similar to Figure 2, in that an encrypted document 310 is passed to a decryption step 312 (which uses a private key 314) and a rendering application 316, resulting in presentation data 318. However, an additional layer of protection is provided by a protecting shell 320. The protecting shell 320 allows the document 310 to be decrypted and rendered without ever leaving clear content (as in the clear content 216 of Figure 2) available to be intercepted. This is accomplished by including decryption and rendering elements within the document 310, as will be described below with reference to Figure 5. The included decryption and rendering elements are adapted to limit the user's interaction with the SPD, prohibiting certain operations (such as saving the document or performing cut-and-paste operations) according to the user's permissions.

Figure 4 is a more sophisticated version. The scheme of Figure 4 includes an intermediate "polarization" step adapted to secure the document after it has been decrypted but before it is rendered. First, the encrypted document contents 410 are

passed to a polarizer 412. The polarizer 412 receives the user's private key 414 and, via a decryption step 416, decrypts the document contents 410. Concurrently, the polarizer 412 receives a polarization key 418 from the user's system.

This polarization key 418 is used by the polarizer 412 to transform the document to a version having polarized contents 420. All of these operations can take place in the open, without any kind of protective mechanism, provided the polarizer 412 does not store a clear version of the document between decrypting it and polarizing it.

In one embodiment of the invention, the polarization key 418 represents a combination of data elements taken from the user's system's internal state, such as the date and time of day, elapsed time since the last keystroke, the processor's speed and serial number, and any other information that can be repeatably derived from the user's system. It is useful to include some time-derived information in the polarization key 418 so that interception and seizure of polarized contents 420 would not be useful. Further rendering of the polarized document would not be possible, as the system time would have changed too much.

Then, once again within a protecting shell 422, the polarized contents 420 are passed to a rendering application 424. As discussed above, typical rendering applications are third-party applications such as Microsoft Word or Adobe Acrobat Reader. However, it is likely that such external rendering applications will not be able to process the polarized contents 420, as the contents, any formatting codes, and other cues used by the renderer will have been scrambled in the polarization process.

Hence, the rendering application 424 must be commutative (or at least fault-tolerant), or it must receive polarized contents 420 that are largely complete and processable by the application. The latter possibility will be discussed below, in connection with Figure 9.

The output of the rendering application is polarized presentation data 426, which has been formatted by the rendering application 424 but is still polarized, and hence not readable by the user. The polarized presentation data 426 is passed to a depolarizer 428, which receives the polarization key 418 and restores the original form of the document as

presentation data 430. In one embodiment of the invention, the depolarization function is combined with the rendering or display function. In this case, the polarized presentation data 426 is received directly by a display device, which can be separate from the user's system and receive data over a communications channel.

Creation of the polarization key 418, the rendering application 418, and the depolarization step 428 are all elements of the protecting shell 422; these are tamper-resistant program elements. It is contemplated that all computational (or transformation) steps that occur within the protecting shell 422 will use local data only, and will not store temporary data to any globally accessible storage medium or memory area; only the explicit results will be exported from the protecting shell 422. This approach will prevent users from easily modifying operating system entry points or scavenging system resources so as to intercept and utilize intermediate data.

It should be noted that the presentation data 430 of Figure 4, in alternative embodiments of the invention, can be either device independent or device dependent. In the device-independent case, additional processing by a device driver (such as a display driver or a printer driver) typically is necessary to complete the rendering process. In the presently preferred device-dependent case, the device-specific modifications to the presentation data have already been made (either in the rendering application 424 or the depolarizing step 428), and the presentation data 430 can be sent directly to the desired output device.

The decryption schemes described with reference to Figures 3 and 4 above are enabled by a unique document structure, which is shown in detail in Figure 5. As discussed above, certain operations performed by the system and method of the invention require trusted components. One way to ensure that certain unmodified code is being used to perform the trusted aspects of the invention is to provide the code along with the documents. The various components of a self-protecting document according to the invention are illustrated in Figure 5.

The problem of document protection is approached by the invention without any assumptions on the presence of trusted hardware units or software modules in the user's

system. This is accomplished by enhancing a document to be an active meta document object. Content owners (i.e., authors or publishers) attach rights to a document that specify the types of uses, the necessary authorizations and the associated fees, and a software module that enforces the permissions granted to the user. This combination of the document, the associated rights, and the attached software modules that enforce the rights is the self-protecting document ("SPD") of the invention. A self-protecting document prevents the unauthorized and uncontrolled use and distribution of the document, thereby protecting the rights of the content owners.

The self-protecting document 510 includes three major functional segments: an executable code segment 512 contains certain portions of executable code necessary to enable the user to use the encrypted document; a rights and permissions segment 514 contains data structures representative of the various levels of access that are to be permitted to various users; and a content segment 516 includes the encrypted content 116 (Figure 1) sought to be viewed by the user.

In a preferred embodiment of the invention, the content segment 516 of the SPD 510 includes three subsections: document meta-information 518 (including but not limited to the document's title, format, and revision date), rights label information 520 (such as a copyright notice aitached to the text, as well as rights and permissions information), and the protected content 520 (the encrypted document itself).

In one embodiment of the invention, the rights and permissions segment 514 includes information on each authorized user's specific rights. A list of terms and conditions may be attached to each usage right. For example, user John Doe may be given the right to view a particular document and to print it twice, at a cost of $10. In this case, the rights and permissions segment 514 identifies John Doe, associates two rights with him (a viewing right and a printing right), and specifies terms and conditions including the price ($10) and a limitation on printing (twice). The rights and permissions segment 514 may also include information on other users.

In an alternative embodiment, the rights and permissions segment 514 includes only a link to external information specifying rights information. In such a case, the

actual rights and permissions are stored elsewhere, for example on a networked permission server, which must be queried each time the document is to be used. This approach provides the advantage that rights and permissions may be updated dynamically by the content owners. For example, the price for a view may be increased, or a user's rights may be terminated if unauthorized use has been detected.

In either scenario, the rights and permissions segment 514 is cryptographically signed (by methods known in the art) to prevent tampering with the specified rights and permissions; it may also be encrypted to prevent the user from directly viewing the rights and permissions of himself and others.

The executable code segment 512, also called the "SPD Control," also contains several subsections, each of which comprises a software module at least partially within the executable code segment. In one embodiment of the invention, the Java programming language is used for the SPD Control; however, it is contemplated that any platform-independent or platform-specific language, either interpreted or compiled, can be used in an implementation of this invention.

A rights enforcer 524 is present to verify the user's identity, to compare a requested action by the user to those actions enumerated in the rights and permissions segment 514, and to permit or deny the requested action depending on the specified rights. The operation of the rights enforcer 524 will be discussed in further detail below, in connection with Figure 7.

A secured polarization engine 526 is also present within the executable code segment 512; it serves to read and polarize the data according to the system state (or other polarization key) as discussed above. In a preferred embodiment of the invention, the polarization engine 526 acts upon the document before it is stored or decrypted, so the document is never stored in the clear on the user's system. The polarization engine 526 is secured, that is, it is cryptographically signed and encrypted, to prevent tampering, reverse-engineering, and disassembling.

A counterpart depolarization engine 528 is also included to enable the generation of clear presentation data from the polarized content (see Figure 4). The depolarization

engine includes a set of secure window objects, providing a relatively tamper-proof interface to the rendering API (application program interface) of the user's system. The secure window objects are resistant to being intercepted, thereby reducing the possibility that the document, in its clear form, can be reconstructed by intercepting and receiving the data intended for the operating system.

A counterpart depolarization engine 528 is also included to enable the generation of clear presentation data from the polarized content (see Figure 4). The depolarization engine 528 provides a relatively tamper-proof interface to the logical or physical output device (e.g., the user's display device). The input to the depolarization engine 528 is polarized presentation data. Therefore, if that data is intercepted, it will not reveal any of the clear content without further depolarization which depends on, for example, the user's system state.

A secure viewer 530 is optionally included in the executable code segment 512. The secure viewer 530 is used to permit only those levels of access that are permitted according to the rights and permissions segment 514. For example, if the user purchased only sufficient rights to view a document (and not to save or print it), the viewer will not permit the user to save, print, or perform the standard cut-and-paste operations possible in most modern operating systems.

Finally, a rendering engine 532 is included or referenced within the executable code segment 512. The rendering engine 532 need not be secure. Accordingly, the code for the rendering engine 532 can be included within the SPD applet, or alternatively retrieved (via a secure link) from some other location. In either case, the rendering engine 532 is adapted to receive polarized document contents and produced polarized presentation data therefrom (see Figure 4).

The foregoing aspects and elements of the self-protecting document 510 will be discussed in further detail below, in conjunction with the operation of the system.

Figure 6 shows the steps performed when a self-protecting document 510 is created and distributed. A generic SPD 610 includes no user-specific rights information and is not encrypted for any particular user. The generic SPD 610 is created from three

items: the original document content 612, in clear (unencrypted) form; a high-level rights specification 614; and an optional watermark 616.

The content 612 is pre-processed (step 618) to lay out the document as desired by the author or publisher. For example, a preferred page size, font, and page layout may be selected. The content 612 is essentially "pre-rendered" in the content pre-processing step so that it will be in a format that is compatible with users' systems and the SPD. For example, the content 612 may be converted from Microsoft Word (".DOC") or Adobe Acrobat (".PDF") format to a different format specially adapted to be read by the rendering engine 532 (Figure 5). In one embodiment of the invention, multiple versions of the content 612 are generated by the content pre-processing step and stored in the generic SPD 610; those different versions may then be separately purchased by the user according to his needs.

The high-level rights specification 614 sets forth what combinations of access rights are permissible. Such a rights specification is tailored to a particular document, and is capable of describing different groups of rights for different classes of downstream users. For example, a publisher may be given the right to distribute up to 100,000 copies of a document at a $1.00 per copy royalty, with additional copies yielding a $2.00 royalty. Similarly, users may be given the option to purchase a version of the document that "times out" after one month, one year, or never. Several possible limitations are described with reference to a detailed example, which is set forth below.

Digital Property Rights Language (DPRL) is a language that can be used to specify rights for digital works. It provides a mechanism in which different terms and conditions can be specified and enforced for rights. Rights specifications are represented as statements in DPRL. For details, see, for example, U.S. Patent No. 5,715,403 to Stefik, entitled "System for Controlling the Distribution and Use of Digital Works Having Attached Usage Rights Where the Usage Rights are Defined by a Usage Rights Grammar." Enforcement of rights and verification of conditions associated with rights is performed using the SPD technology.

Different rights can be specified for different parts of a digital work using a "work" specification. Within a work specification, different sets of rights applicable to this work are specified. Rights can be grouped into named-groups called "rights groups". Each right within a rights group is associated with a set of conditions. Conditions can be of different types: fee to be paid, time of use, type of access, type of watermark, type of device on which the operation can be performed, and so on. DPRL allows different categories of rights: transfer, render rights, derivative work rights, file management rights and configuration rights. Transport rights govern the movement of a work from one repository to another. Render rights govern the printing and display of a work, or more generally, the transmission of a work through a transducer to an external medium (this includes the "export" right, which can be used to make copies in the clear). Derivative work rights govern the reuse of a work in creating new works. File management rights govern making and restoring backup copies. Finally, configuration rights refer to the installation of software in repositories.

An exemplary work specification in DPRL is set forth below:

(Work:
(Rights-Language-Version: 1.02)
(Work-ID: "ISDN-1-55860-166-X; AAP-2348957tut")
(Description: "Title: 'Zuke-Zack, the Moby Dog Story'
Author: 'John Beagle'
Copyright 1994 Jones Publishing")
(Owner: (Certificate:
(Authority: "Library of Congress")
(ID: "Murphy Publishers")))
(Parts: "Photo-Celebshots-Dogs-23487gfj" "Dog-Breeds-Chart-AKC")
(Comment: "Rights edited by Pete Jones, June 1996.")
(Contents: (From: 1) (To: 16636))
(Rights-Group: "Regular"
(Comment: "This set of rights is used for standard retail editions.")

```
(Bundle:
(Time: (Until: 1998/01/01 0:01))
        (Fee: (To: "Jones-PBLSH-18546789")(House: "Visa")))
(Play:
        (Fee: (Metered: (Rate: 1.00 USD) (Per: 1:0:0) (By: 0:0:1))))
(Print:
        (Fee: (Per-Use: 10.00 USD))
        (Printer:
                (Certificate:
                        (Authority: "DPT"
                        (Type: "TrustedPrinter-6")))
        (Watermark:
                (Watermark-Str: "Title: 'Zeke Zack - the Moby Dog' Copyright
                1994 by Zeke Jones.  All Rights Reserved.")
                (Watermark-Tokens: user-id institution-location render-name
                render-time))))
(Transfer: )
(Copy: (Fee:   (Per-Use: 10.00 USD)))
(Copy: (Access:
                (User:  (Certificate:
                                (Authority: "Murphy Publishers")
                                (Type: "Distributor")))))
(Delete:)
(Backup:)
(Restore:       (Fee:   (Per-Use: 5.00 USD)))))
```

This work specification has a rights group called "Regular," which specifies rights for standard retail editions of a book titled "Zuke-Zack, the Moby Dog Story." The work specification expresses conditions for several rights: play, print, transfer, copy, delete,

backup, and restore. The work in the example includes two other parts, a photograph and a chart of breeds incorporated from other sources. A "bundle" specification bundles a set of common conditions that apply to all rights in the group. This specification states that all rights in the group are valid until January 1, 1998 and that the fee should be paid to account "Jones-PBLSH-18546789". The clearing-house for this transaction should be Visa. The following contract applies: the work can be played by paying $1.00 every hour, where fee is accumulated by the second; the work can be printed on TrustedPrinter-6 which is certified by "DPT" for a fee of $10.00 per print; the printed copy should have a watermark string (as depicted) and a list of tokens signifying "fingerprint" information known at the time it is printed; this work can be copied either by paying $10.00 or by acquiring a distributor certificate from Murphy publishing; and unrestricted transfer, deletion or backing up of this work is permitted (restoration costs $5.00).

The high-level rights specification 614 is also subject to a pre-processing step (step 620), in which the high-level (i.e., human-readable) specification is compiled into a more-efficient data structure representation for use by the invention.

The generic SPD 610 is then created (step 622) by combining the pre-processed content 612, the pre-processed rights specification 614, and the watermark 616. A watermark may be added by any means known in the art; it may be either visible or concealed within the SPD. The generic SPD 610 may also optionally be encrypted by the author/publisher 110 for transmission to the distributor 114 (Figure 1).

The generic SPD 610 is then received by the distributor 114, and is stored for later customization. When a user request 624 is received by the distributor 114 (either directly or through the clearinghouse 122 or other intermediary), the distributor 114 creates a set of user permissions (step 626) that is consistent with both the user request 624 and the rights specification 614. If there is no such consistent set of permissions, then no further action is performed on that user's behalf (other than an optional notification message to the user).

The user permissions and the user's public key 628 are then used to generate (step 630) a customized SPD 632 adapted to be used by the user. The user permissions from

step 626 are stored in the rights and permissions segment 514 of the SPD 632, and the user's public key 628 is used to encrypt the content in the content segment 516 of the SPD 632. A public-key encryption mechanism can be used to transform the SPD from the generic form to the customized SPD 632. Such a mechanism is useful if the SPD has to be confidentially transferred between different parties, e.g., author to publisher to retailer to consumer, with rights protection at each stage. It should further be noted that multiple user requests can be composed and accommodated within a single SPD 632; there are techniques known in the art that are capable of using multiple public keys to encrypt a document such that any of the users' private keys can be used to decrypt it.

The resulting custom SPD 632 is then transmitted to the user 118 by any available means, such as via a computer network or stored on a physical medium (such as a magnetic or optical disk).

The operations performed when a user receives an SPD are depicted in the flow diagram of Figure 7. The SPD is first received and stored at the user's system (step 710); in many cases, it is not necessary to use the SPD right away. When usage is desired, the user is first authenticated (step 712), typically with a user name and a password or key. The system then determines what action is desired by the user (step 714). When an action is chosen, the rights-enforcement step of the invention (step 716) verifies the conditions associated with the desired action (such as the fee, time, level of access, watermark, or other conditions); this can be performed locally via the SPD applet 512 (Figure 5) or by accessing a rights enforcement server.

If the rights enforcement step (step 716) fails, an update procedure (step 718) is undertaken. The user may choose to update his permissions, for example by authorizing additional fees. After the satisfactory verification of conditions, a pre-audit procedure (step 718) is performed, in which the SPD system logs verification status to a tracking service (e.g., the audit server 130 of Figure 1). The content is then securely rendered to the screen (step 722) as discussed above. When the user is finished, a post-audit procedure (step 724) is performed in which the amount of usage is updated with the tracking service. The SPD system then awaits further action.

The protection yielded by the SPD is derived from the user's inability to capture a useful form of the document at any intermediate stage during the rendering process. This is accomplished by decrypting the document contents to a clear form at the latest possible stage, ideally in the last step.

The SPD decryption model is illustrated in Figure 8. E denotes the encryption function performed by the publisher; D denotes the decryption performed at the user's system, and R denotes the rendering transformation. Many prior systems use a first sequence of transformations 810, D(E(x)) followed by R(D(E(x))). As stated previously, the early decryption leaves the document in a vulnerable state. Ideally, the transformations are performed in the reverse order 812, R′(E(x)) followed by D(R′(E(x))). This postpones decryption to the latest possible time.

The existence of R′, a rendering operation that can be performed before decryption, is determined by the following equality:

$$D(R'(E(x))) = R(D(E(x)))$$

In case that the encryption and decryption functions are commutative, that is, E(D(x)) = D(E(x)) for any x, the existence of R′ is ensured:

$$R'(y) = E(R(D(y))) \text{ for } y = E(x)$$

In practice, encryption and decryption functions in popular public-key cryptographic systems such as the RSA system and ElGamal discrete logarithm system satisfy the commutation requirement. This means that the transformation R′ exists if these cryptographic systems are used for encryption and decryption.

The path $x' = D(R'(E(x)))$ portrays an ideal SPD solution to the document protection against unauthorized document usage and distribution. A scenario of distributing and using a document can be described as follows. When a user purchases the document, the document is encrypted using a user's public information and is transmitted over an insecure network channel such as the Internet. The encrypted document has the rights information attached to it and a protecting applet 512 that enforces the rights and permissions granted to the user by the content owner. Upon a user's request on using the document, the applet verifies the rights and permissions and

generates from the encrypted document the presentation format of the original document. As any intermediate form of the document before the final presentation data is encrypted with the user's private information, the SPD model of document protection ensures that any intermediate form of the document is not useful to other systems wherever it is intercepted.

Clearly, this ideal model relies on whether or not the transformation R' that corresponds to the rendering transformation R can be computed efficiently, and in particular on whether or not an invocation of the decryption function D is necessary during an implementation of R'. A trivial case in which R' can be implemented efficiently is where R is commutative with the encryption function E. When this happens,

$$R'(y) = E(R(D(y))) = R(E(D(y))) = R(y)$$

for $y = E(x)$. In this case, $R' = R$.

Consideration of Figure 8 reveals that many intermediate solutions (e.g., intermediate solutions 814, 816, and 818) to the document protection problem may exist on the user's system between the two extremes $x' = R(D(E(x)))$, which has no protection on $x = D(E(x))$, and $x' = D(R'(E(x)))$, which has ideal protection (under the assumptions set forth above). As depicted in Figure 8, one may consider different paths from the encrypted document E(x) to the presentation data x' that correspond to different combinations of partial rendering transformations and partial decryption transformations. Again, it should be recognized that delaying the decryption D in any path increases the protection level to the document.

As discussed above, one alternative method of delaying decryption to the last possible moment employs a polarization technique that encrypts only the document contents, not the format or the entire document as a whole. This possibility is shown in Figure 9. Beginning with the clear document content 910 (which, it should be noted, does not exist in any single identifiable location during the user's processing, but is rather a transient state occurring within step 412 of Figure 4), the document is split (step 912) into a data portion 914 and a format portion 916. The data portion 914 is polarized (step

918) using the polarization key 920 and merged (step 922) with the clear format portion 916. This results in polarized content 924 that can be rendered to polarized presentation data without first decrypting the content. It should be observed that this form of polarization is likely less secure than wholesale encryption with the polarization key, since a lot of information can potentially be derived from the layout of a document, word lengths, line lengths, etc.; however, this scheme will present a useful deterrent to casual copyright infringement.

A method of protecting a digital work during replay which employs a blind transformation function is shown with reference to Figure 10. In Figure 10, an encrypted digital work 1010 is provided to replay application 1012. Digital work 1010 has been encrypted with a format preserving encryption scheme which enables replay application 1012 to generate encrypted presentation data 1016. Encrypted presentation data 1016 is then sent to decryption engine 1018 where it is decrypted into clear presentation data 1020. Presentation data is now in the clear, but less likely to be regenerated into the original digital form. If presentation data 1020 can be viewed or used directly by the user, then no further processing is required. However, sometimes an additional rendering is required by a display system such as a printer. In such a case, presentation data 1020 is provided to the display system's rendering application (in the case of a printer this could be a decomposer) 1022 which generated image data 1024. Image data 1024 is then provided to display device 1026.

In a general context, the problem of blind transformation can be stated as follows. Suppose a client Cathy wants a server Steve to compute for her a function value F(a,x) with his (public or private) data a and her private data x, and Cathy wishes, for privacy concerns, that the transformation is done without Steve knowing her private data x and the function value F(a,x). From Steve's point of view, this means that he computes F(a,x) for Cathy but with his eyes blindfolded. What this means is that Cathy would like the server Steve to perform the transformation only with data $E_k(x)$ encrypted using Cathy's key k, and return to her the function value $E_k(F(a,x))$ again encrypted using her key k. If Steve can perform the transformation using encrypted data, then Cathy has avoided

disclosing the data x in the clear and the result F(a,x) in the clear. The ideal model of blind transformation with partially encrypted data is shown below:

$$\begin{array}{ccc} (a,x) & \xrightarrow{E_k} & (a,E(x)) \\ F\downarrow & & \downarrow F' \\ F(a,x) & \xleftarrow[D_{k^{-1}}]{} & F'(a,E(x)) \end{array}$$

The function F′ that makes the diagram commute is what Steve really computes, and the transformation result $F'(a,E_k(x)) = E_k(F(a,x))$ is ready for decryption to reveal the desired function value F(a,x). As Steve does not "see" the clear data x as well as the function value F(a, x), he carries out a "blind" transformation for Cathy.

A protocol for blind transformation can be described as follows for the blind evaluation of the function F(a,x):

(i) Cathy encrypts x using her encryption key k, resulting $E_k(x)$.

(ii) Cathy sends $E_k(x)$ to Steve.

(iii) Steve evaluates the modified version F′ of the function F at the clear data a and encrypted data $E_k(x)$.

(iv) Steve returns the result $F'(a,E_k(x))$ back to Cathy.

(v) Cathy decrypts $F'(a,E_k(x))$ using her decryption key k^{-1} and obtains F(a,x).

The ideal model of blind transformation introduced here can be regarded as a generalization of blind signatures and instance hiding. Blind transformation now allows partially encrypted data as input and, more importantly, it permits the function F′ that the server computes to be possibly different from the intended function F. By computing F′ instead of F, the server, though still blindfolded, is aware of the input being partially encrypted and hence is cooperative with the client. The blind transformation and secure mobile computing share a common goal in keeping the function value that the server computes private to the client, but they differ in that the client supplies the data input and the server supplies (a program that evaluates) the function in blind transformation, while it is the other way around in secure mobile computing. Note that blind transformation allows some portion of the data (e.g., a) to be in clear. This enables use of some dynamic

yet clear data in the rendering process, such as display window size, reference positions for shifting content, scaling factor and coefficients in a rotation operation.

Blind transformation works only if there exist functions F and F' to compute the encrypted data. It can be shown that multivariate, integer coefficient affine functions using additive encryption schemes permit many document rendering functions of the affine type on the x- and y-coordinates to be evaluated in blind transformation. For a given encryption scheme S, a function $F : X \to X$ is said to be *S-blindly computable* if there exists some function $F' : X \to X$ such that the computational complexity for evaluating F' is a polynomial of the one for evaluating F, and

$$F(a,x) = D^{k-1}(F'(a,E_k(x)))$$

for any $k \in K$ and $x \in X$. A function $F: X \to X$ is said to be *blindly computable* if there exists an encryption scheme S with X being a subset of its message space such that F is S-blindly computable.

Any multivariate, integer-coefficient affine function is S-blindly computable for any additive encryption scheme. Specifically, let $F_{x0;a1,\ldots,ak}(x_1, \ldots, x_k) = x_0 + \sum_{i=1}^{k} a_i x_i$ be a multivariate affine function with a constant $x_0 \in X$, integer coefficients a_i and variables $x_1, \ldots x_k$ in X. Then, for any key $k \in K$, there exists a computationally efficient function

$$F'_{y_0,b_1,\ldots,b_k}(y_1,\ldots,y_k) = y_0 \oplus \bigoplus_{i=1}^{k} b_i y_i \text{ such that}$$

$$E_k(F_{x_0,a_1,\ldots,a_k}(x_1,\ldots,x_k)) = E_k(x_0 + \sum_{i=1}^{k} a_i x_i) = F'_{y_0,b_1,\ldots,b_k}(E_k(x_k)).$$

Indeed, the constant y_0 and integer coefficients b_i in $F'_{y_0,b_1,\ldots,b_k}$ can be taken to be $y_0 = E_k(x_0)$, $b_i = a_i$, $i = 1, \ldots, k$. The blind transformation of multivariate, integer coefficient affine functions using additive encryption schemes allows many document rendering functions of the affine type on the x-and y-coordinates to be evaluated in the blind manner, providing a theoretical foundation for the format-preserving encryption and trusted rendering of documents described herein.

A document is usually a message that conforms to a certain format. For document encryption, in addition to simply encrypting the entire document, there are many different ways to encrypt only some parts of the document. The goal here is that the information leakage about the unencrypted portion cannot be used, or if it does leak, it is computationally difficult to reconstruct the clear, original document.

If an encryption scheme which preserves formatting information of the digital work, then any transformation function (replay application or rendering application) may be used. An example of a format preserving encryption method is described for convenience with reference to token-based documents. The method for format-preserving encryption can be easily extended or applied to documents in other formats (such as HTML/XML, Microsoft WORD, Acrobat PDF, etc.). In a token-based format such as the Xerox DigiPaper, each page image of a document is represented as a "dictionary" of token images (such as characters and graphics elements) and location information (indicating where those token images appear in the page). Thus, multiple occurrences of the same token in the document can be represented using just a single image of that token in the dictionary.

The process of rendering a document in such a format is then accomplished by consecutively reading in token locations, retrieving images of the tokens from the dictionary and drawing the images at the specified locations. The benefits of token-based documents are compact file size and fast rendering speed for use in distributing, viewing and printing of electronic documents. In the DigiPaper format, tokens are stored as binary images using the CCITT Group 4 compression format, or as color images using JPEG compression, and the position information of the tokens is further compressed using Huffman coding.

For convenience, a token-based document D of P pages is formally modeled as a table (dictionary) of tokens T of size $|T|$, together with a sequence of P tables of locations L_i of size $|L_i|$ ($1 \le i \le P$), representing the P page images. Each entry T[j], $1 \le j \le |T|$, is a pair (id[j],t[j]) of an identifier id[j] and an image t[j] of the j-th token. Each entry $L_i[k]$, $1 \le k \le |L_i|$, in the i-th image location table L_i is a triple (id[k],x[k],y[k]) representing the k-

th token occurrence in the i-th page image, where id[k] is the token identifier, and x[k], y[k] are its x and y-coordinate differences from the previous (k -1)-th token occurrence in the page. For example, take the simple document shown in Figure 11. The token dictionary and location table (using x, y coordinates) for this document are shown in Figures 12 and 13 respectively.

The schematic pseudo-code Render(D) below shows how page images of a document D are rendered. In the code, x_0, y_0 are the base references for the x- and y-coordinates for each page, Lookup(T,id[k]) is a subroutine that, upon the input of the dictionary T and a token identifier id[k], returns a token image t in T corresponding to the given identifier, and Draw(x,y,t) is a subroutine that draws the token image t at the location (x,y).

```
Render(D)
{
    Load T into memory
    for i = 1 to P do
    {
        Load L_i into memory
        x = x_0
        y = y_0
        for k = 1 to |L_i| do
        {
            x = x + x[k]
            y = y + y[k]
            t = Lookup(T,id[k])
            Draw(x,y,t)
        }
    }
}
```

In addition to the shifting transformation $x' = x + a$, $y' = y + b$ as used in the schematic rendering process described above, there are several other coordinate transformations that may occur during the document rendering.

Scaling. The scaling transformation is of the form $x' = ax$, $y' = by$, where a and b are scaling factors for the x-coordinate and y-coordinate, respectively. Scaling may be caused by resizing the display window or print paper.

Rotation. The rotation transformation is $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ for some constants a, b, c, d, which form a 2-by-2 rotation matrix. This transformation is needed when the page image is rotated.

Affine Transformation. An affine transformation is one of the form $x = ax + by + e$; $y = cx + dy + f$ for some constants a, b, c, d, e, f. In the vector form, it is: $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$. Clearly, shifting, scaling and rotation transformations are special cases of affine transformations. It is those affine type transformations that make it possible to achieve a high-level trusted rendering under encryption of coordinate information using additive encryption schemes described below.

A special class of encryption schemes, namely, additive encryption schemes, are used to carry out blind transformation of functions of the affine type, which provides a foundation for trusted rendering of documents. Blind transformation by a rendering transformation R and R′ of an encrypted document satisfies the relationship: $D(R'(E(x))) = R(D(E(x)))$, where E is an encryption function and D is a decryption function for E. If E(x) is an additive encryption scheme, then $R' = R$.

An encryption scheme S generally consists of basically five components: (i) a message space X which is a collection of possible messages, (ii) a ciphertext space Y which is a collection of possible encrypted messages, (iii) a key space K which is a set of possible keys, (iv) a computationally efficient encryption function $E : K \times X \rightarrow Y$ and (v) a computationally efficient decryption function $D : K \times Y \rightarrow X$. For each key $k \in K$, there is a unique key $k^{-1} \in K$, such that the encryption function $E_k = E(k,) : X \rightarrow Y$ and

decryption function $D_{k^{-1}} = D(k^{-1},) : Y \rightarrow X$ satisfy that, for every message $x \in X$, $D_{k^{-1}}(E_k(x)) = x$. The key k is called an encryption key and k^{-1} its corresponding decryption key.

Such defined encryption schemes can be varied in several ways to cover a wide range of concrete encryption schemes used in practice. One variation is to consider whether or not keys used for encryption and decryption are different. In the case where all encryption keys k are same as their corresponding decryption keys k^{-1}, the scheme is a symmetric (or private-key) one; otherwise, the scheme is asymmetric. In the case where, for all possible k, k^{-1} is different from k and computationally difficult to derive from k, the scheme is a public-key encryption scheme.

Another variation is to differentiate deterministic and probabilistic encryption schemes. In a deterministic scheme, all the encryption and decryption functions E_k and D_{k-1} are deterministic functions, while in a probabilistic scheme the encryption function E_k can be non-deterministic, namely, applying the function to a message twice may result in two different encrypted messages.

An additive encryption scheme is an encryption scheme whose message space X and ciphertext space Y possess some additive structures and encryption function $E_k = E(k,) : X \rightarrow Y$ is homomorphic with respect to the additive structures. Specifically, let X = (X, +, 0) and Y = (Y,⊕, 0) be two commutative semigroups with (possibly different) zero elements 0 satisfying, for example, for all x, x + 0 = x and 0 + x = x, and efficient operations + and ⊕. An encryption scheme is said to be *additive* if, for any $k \in K$ and any $x, x' \in X$, $E_k(x + x') = E_k(x) \oplus E_k(x')$, and the operation ⊕ does not reveal the clear messages x and x′. The last condition on ⊕ makes additive encryption schemes non-trivial. Without this condition, the operation ⊕ on Y can be trivially defined $y \oplus y' = E_k(D_{k-1}(y) + D_{k-1}(y'))$; that is, it is accomplished by first decrypting the arguments, then adding them together and finally re-encrypting the result.

Closely related to additive encryption schemes are multiplicative ones. An encryption scheme is said to be *multiplicative* if its spaces X and Y have the ring

structures (i.e., in addition to their additive structures, they have respective multiplications $\times$ and $\otimes$ that are distributive over their additions $+$ and $\oplus$, and multiplicative identities), the encryption function E_k is homomorphic with respect to the multiplications, $E_k(x \times x') = E_k(x) \otimes E_k(x')$; and the operation $\otimes$ does not reveal the clear messages x and x'.

In general, additive (as well as multiplicative) encryption schemes are not non-malleable, since a non-malleable scheme requires that, given an encrypted message it is (at least computationally) impossible to generate a different encrypted message so that the respective clear messages are related. Accordingly, they have a weakness against active attacks where the adversary attempts to delete, add or alter in some other way the encrypted messages. However, when these schemes are used to encrypt documents, extra measures in data integrity and message authentication can be taken to reduce risks caused by these active attacks on document integrity as well as confidentiality. Moreover, end users are less motivated to initiate active attacks, as the attacks will affect document contents that the users are going to use and consume.

Not all encryption schemes can be defined as additive ones in an easy and natural manner. In fact, some encryption schemes are designed with a requirement of being non-additive or at least being able to convert into non-additive. Nevertheless, there are many examples of additive encryption schemes that can be used in the method of format-preserving encryption and trusted document rendering. **Mult**, **Exp** and **EG** (three deterministic schemes), **OU** (probabilistic) and **RSA** are examples of additive encryption schemes (with varying degrees of vulnerability to attack) may be used in the format preserving method.

Multiplicative Cipher (**Mult**) is a symmetric encryption scheme, where $X = Y = Z_n = \{0, 1, \ldots n-1\}$ for some integer $n > 0$. The encryption of a message x using a key a is

$$y = E_a(x) = ax(\mathrm{mod}\ n)$$

and the decryption of a message y using a key a is

$$x = D_a(y) = a^{-1}y(\mathrm{mod}\ n),$$

where a^{-1} is the multiplicative inverse of a modulo n.

Exponential Cipher (**Exp**) is a symmetric cipher, where $X = Z_{p-1}$ and the ciphertext space $Y = Z_p$ for some prime p, and K is the set of all generators of the multiplicative group Z^*_p. For any generator $g \in K$, the encryption function is defined as the exponential function

$$E_g(x) = g^x \pmod p,$$

while the decryption function is defined as the logarithm function

$$D_g(y) = \log_g y \pmod{(p-1)}.$$

Semi-probabilistic ElGamal Cipher (**EG**) extends the exponential cipher to the ElGamal cipher, which leads the ElGamal cipher to run in a semi-probabilistic mode. For each message $x \in Z_p$, where $Z_p = \{1, \ldots p-1\}$ for some prime p, g is a generator in the multiplicative group Z^*_p, the private decryption key for a user is a random number $a \in Z^*_{p-1}$, the public encryption key $\alpha = g^a \pmod p \in Z_p$, the encryption $E_\alpha(x, r)$ depends on a uniformly chosen random number $r \in Z^*_{p-1}$:

$$E_\alpha(x,r) = (g^r \pmod p, x\alpha^r \pmod p) = (s,t).$$

For an encrypted message (s, t), the decryption function is defined as

$$D_\alpha(s,t) = t(s^a)^{-1} \pmod p.$$

The ElGamal cipher in its original form as described above is hardly additive. However, the operator $\oplus$ can be partially defined on the ciphertext of those x's that share a same random number r, as follows:

$$E_\alpha(x, r) \oplus E_\alpha(x', r) = (s, t) \oplus (s, t') = (s, t + t') = E_\alpha(x + x' \pmod p, r).$$

This partially defined operation is applicable when a batch of messages are encrypted using a same random number r.

Okamoto-Uchiyama Cipher (**OU**). Okamoto and Uchiyama proposed an additive, public-key encryption scheme in T. Okamoto and S. Uchiyama. "A New Public-Key Cryptosystem as Secure as Factoring", *Eurocrypt'98*, Lecture Notes in Computer Science 1403, 308-318, 1998, which is probabilistic and provably as secure as the intractability of factoring $n = p^2q$ against passive adversaries. Choose two large primes p, q of k bits for

some $k > 0$,and let $n = p^2q$. Choose $g \in Z^*_n$ at random such that the order of $g_p = g^{p-1} \pmod{p^2}$ is p. Let $h = g^n \pmod{n}$. The message space X of the OU scheme is the set Z^*_p (not the set $\{1, \ldots 2^{k-1}\}$ as claimed by Okamoto and Uchiyama) and the ciphertext space Y is Z_n. For a user, a public key is a tuple (n, g, h, k) and its corresponding private key is the pair (p, q) of the primes. To encrypt a message $x \in X$, a random number $r \in Z_n$ is chosen uniformly. Then the encrypted message is

$$y = E_{(n,g,h,k)}(x,r) = g^x h^r \pmod{n}.$$

To decrypt the encrypted message y, a "logarithmic" function $L : \Gamma \rightarrow \Gamma$,

$$L(x) = (x - 1)p^{-1} \pmod{p^2}$$

is used, where Γ is the p-Sylow subgroup of Z^*_{p2}, i.e., $\Gamma = \{x \in Z^*_{p2} \mid x \equiv 1 \pmod{p}\}$. With the function L, the decryption function is

$$x = D_{p,q}(y) = L(y^{p-1} \pmod{p^2}) L(g_p)^{-1} \pmod{p^2}.$$

New additive encryption schemes can be constructed from existing ones via the composition construction of encryption schemes. The composition construction can also be used to construct additive encryption schemes from non-additive ones. For instance, the composition of the exponential cipher Exp and any multiplicative encryption scheme S (such as RSA) results in an additive one.

Additive encryption schemes enable blind transformation with partially encrypted data, which serves a foundation for trusted rendering of documents, as discussed above. In particular, additive encryption schemes can be used to perform blind transformation of affine functions with clear coefficients and encrypted variables.

Returning to the example of a token-based document, since a token-based document D consists of a dictionary T of token images and a sequence of location tables L_i (one for each page image), the idea is to encrypt the content of the dictionary T and location tables L_i, resulting in a dictionary T' of encrypted token images and tables L'_i of encrypted locations. Recall that the dictionary T consists of a collection of pairs (id[j], t[j]), $j = 1, \ldots |T|$. Associated with T is a subroutine Lookup in the rendering process that, given a valid token identifier id, returns its corresponding token image t in T. In encrypting the dictionary T, there are three basic choices: encrypting token identifiers,

token images, or both. Encrypting either identifiers or token images helps unlink the connection between the identifiers and their token images. In addition, encrypting token images protects proprietary token images. In any case, it is desirable to allow valid access to the dictionary only within the rendering process P, while making it computationally difficult to obtain a copy of the entire, clear contents of the dictionary. This is possible because in many cases the valid identifiers (e.g., Huffman codewords) are only a very small subset of all binary strings of up to a certain length, and consequently any exhaustive identifier search will not be efficient.

More formally, given the dictionary T and the Lookup subroutine that accesses it, the requirement on encrypting the dictionary is that the encrypted dictionary T′ and the corresponding subroutine Lookup′ satisfy the following constraints:

(1) For any encrypted identifier $E_k(id)$, $Lookup'(T',E_k(id)) = E_k(Lookup(T,id))$ and

(2) Given T′ and Lookup′, it is computationally infeasible to reconstruct T.

For an encryption scheme S, T′ and Lookup′ can be constructed as follows. Let ID be the set of all syntactically possible identifiers; in particular, $ID^* \subseteq ID$, where $ID^* = \{id \mid (id,t) \in T\}$. Let h be a one-way hash function whose domain is ID. Then the encrypted token dictionary T′ is derived from T as follows: for every (id,t) pair in T, a pair $(h(id),E_k(t))$ is inserted into T′. The modified subroutine Lookup′ uses the algorithm:

```
Lookup'(T',id)
{
        id' = h(id)
        t' = Lookup(T',id')
        return (t')
}
```

Notice that the return value of Lookup′ is an encrypted token image. The decryption of this image will be postponed to into the final subroutine Draw′ in the rendering process, which is part of the trusted rendering described below.

This dictionary encryption is computationally feasible, both in terms of storage space overhead and in terms of running-time overhead, to compute with encrypted versions of token dictionaries. If the hashing and encryption algorithms used in the Lookup′ subroutine are secure enough, then it is computationally very difficult to recover T given T′ and Lookup′.

Since each entry in a location table L_i consists of an identifier, and location difference in x- and y-coordinates, any combination of the three elements can be encrypted. To encrypt the location information, an additive encryption scheme is recommended to enable applying any rendering transformation of the affine type to the location coordinates. For identifiers, a trade-off between document compression and document protection must be made. In a token-based document, a token identifier is usually a codeword of some coding scheme for the compression purpose. For example, when the Huffman code is used to compress the document, the identifiers are the binary Huffman codewords of the tokens based on their occurrence frequency in the document. In this case, simply using a deterministic encryption scheme to encrypt these identifiers offers no effective protection on them. This is because the scheme does not change the occurrence frequency of each token, and hence anyone can re-count the number of occurrences of the encrypted identifiers to re-construct the Huffman codewords that are the identifiers. Therefore, in order to hide occurrence frequencies of the tokens in the document, it is preferred to use a probabilistic encryption scheme to encrypt the identifiers. However, this will interfere with the optimal encoding carried in the identifiers (codewords) and reduce the document compression ratio. This may be undesirable for token-based documents, as achieving a good document compression is one of the design goals for token-based documents.

A reasonable compromise for encrypting L_i is suggested. Choose an additive encryption scheme S, preferably a probabilistic and asymmetric one like the Okamoto-Uchiyama cipher **OU** if encryption and decryption efficiency is not a big problem. For each entry (id,x,y) in L_i, insert $(id,E_k(x),E_k(y))$ into L'_i. If it is also necessary to encrypt the identifiers, entries like $(E_k(id),E_k(x),E_k(y))$ may be inserted into the location table L'_i.

But in this case, the entries in the encrypted dictionary T′ need to be changed to $(E_k(id),E_k(t))$'s, and the subroutine Lookup′ above also needs to be modified to reflect the change.

With the format-preserving encryption of a token-based document mentioned above, the document content can also be protected during the rendering process. The idea is to delay decryption into Draw′(x,y,t). The rendering process is given shown below.

```
Render(D)
{
    Load T into memory
    for i = 1 to P do
    {
        Load Li into memory
        x = Ek(x0)
        y = Ek(y0)
        for k = 1 to |L| do
        {
            x = x ⊕ x[k]
            y = y ⊕ y[k]
            t = Lookup′(T′,id[k])
            Draw′(x,y,t)
            }
        }
    }
    Draw′(x,y,t)
    {
        x = Dk-1(x)
        y = Dk-1(y)
        t = Dk-1(t)
        Draw(x,y,t)
```

}

During the process, all the coordinate and token image information remains encrypted before calling the subroutine Draw'(x,y,t). This is possible for the coordinate information because the encryption scheme is additive. Consequently, the content protection level and rendering process performance of the rendering process rely on the security strength and computational complexity of the scheme used.

In another embodiment of the invention, a digital work is polarized enabling trusted rendering or replay of the digital work without depolarization of the digital content or the presentation data. In this embodiment, the digital work is the type which includes digital content and resource information (also called a system context). Resource information includes formatting information or other information used by a replay or rendering application to convert the digital work into presentation data.

Polarization is a type of transformation which renders the original content unreadable or unusable. For a digital work w, a polarization scheme T, which uses a seed s, generates a polarized digital work w' according to: $w' = T(w, s)$. The same transformation T may also be used to generate the polarized resource information S' according to $S' = T(S, s)$. In this example, a seed s is used to make reverse engineering of the polarization scheme more difficult.

For example, a document type digital work may be polarized using a simple polarization scheme. In a document, the digital content comprises a series of characters in a particular order or location. If the document is to be displayed on a viewing device, each character must be able to be displayed at a particular location for viewing by a user on the viewing device, such as on a monitor. A coordinate system is required for displaying each character on the monitor, so each character in the document can be displayed on the monitor. The digital content contains coordinate information which is referenced by the monitor's coordinate system. For example, in this paragraph, the letter "F" appears at the top line, indented by five spaces.

A simple polarization scheme for jumbling the text of the above paragraph is to translate the location of the letters with respect to the coordinate system. Each letter in

the paragraph has an (x,y) location. Suppose the location (x,y) of each letter in the above paragraph are polarized using a seed (a,b) from a user's system. The following polarization functions may be used to polarize the above paragraph:

$Y = b^y$, for the vertical axis; and

$X = x/a$, for the horizontal axis.

In this example, the user's device coordinate system must be polarized in order for the replay application to transform the digital content into presentation data, i.e., display the paragraph on the monitor descrambled. The user's device coordinate system must be polarized using the same seed (a, b) to generate a polarized coordinate system. The following transformation functions are used to compute both x and y locations of a given point:

$Y = \log_b(Y)$, for the vertical axis; and

$X = aX$, for the horizontal axis,

where $\log_b$ is the logarithm with base b.

When the replay application obtains the location of a character in the polarized digital work, the location is given by $(X,Y) = (x/a, b^y)$. This value is then applied to the device coordinate system $(X,Y) = (\log_b(Y), aX) = (x,y)$. Thus the correct location of "F" is displayed on the user's monitor. In both cases of polarization, the polarized forms of the resource information and the digital work maintain an inherent association. These complementary polarized forms of the resource information and the digital work result in the basis for a effective mechanism to protect the digital work. While the replay application is able to display the polarized digital work, it is only with the polarized system context that the replay application is able to provide clear presentation data.

While polarization, in general, is not as rigorous a protection as encryption, depending on the sensitivity of the digital work to be protected, different levels of polarization can be used. A sensitive work may require a high level of polarization; a lower valued work may require a weaker type of polarization. If the user's environment is trusted, a lower level of polarization may be used. An advantage to using a lower level of polarization is that it requires fewer system resources to create the polarized digital

work and to render or replay the polarized digital work. The type and quality of the polarization seed may also be used in combination with the polarization scheme to determine the level and strength of the polarization. For example, a more complex polarization seed (such as one containing authorization information from a trusted source or a dynamic seed) will provide a higher level of polarization and strength.

Polarization typically occurs at the distribution or manufacturing location. Digital works are polarized usually prior to distribution to the user or customer using a polarization scheme chosen by the manufacturer or distributor. Resource information to be polarized may also be preselected in advance to delivery. Preferably a seed is used for each polarization scheme. Also preferably, the seed is generated using information provided by the user's system context.

When a user purchases a digital work, the user preferably provides information from the user system in which the user intends to replay the digital work. This information may be used to generate the polarization seed for both the polarized digital work and the polarized resource information (sometimes called the polarized system context). Then the polarized digital work and polarized system context or polarized resource information are provided to the user. Also, typically, but not needed for operation of this embodiment of the invention, the polarized digital work and polarized system context may be encrypted prior to distribution to the user. Decryption of both the polarized digital work and system context may be required prior to replay of the polarized digital work into presentation data, depending on the encryption scheme used.

The process for creating a polarized digital work is divided into three steps. These steps are generation of the polarization seed, polarization of the digital work and, polarization of the resource information. Once the polarization seed is generated, the polarization engine is seeded with it. The polarization engine takes as input the digital work or the resource information, and generates the polarized form of the digital work or the resource information based upon the transformation function seeded with the polarization seed. During replay of the polarized digital work, the polarized resource information is utilized to generate the presentation data and/or image data. The same or

different polarization transformation functions can be used for the digital work and the resource information.

A process for creating a polarized digital work is shown with reference to Figure 14. A digital work 1410 includes digital content and a set of resource information used for formatting and rendering the digital content into a form usable or viewable by a user. The digital work 1410 goes through a process of content polarization 1420 in which the digital content is polarized and the resource information is preserved, creating polarized digital work 1422. The content polarization 1420 may occur as shown with reference to Figure 9. A digital work typically includes content, instructions and formatting. While polarization can occur to the entire digital work, preferably only the content is polarized; the instructions and formatting are not polarized. However, in some instances, for some replay applications, some of the resource information contained within the digital work may also be polarized. This is similar for the format preserving encryption method described above.

Resource extraction 1412 extracts at least one resource information from the set of resource information associated with digital work 1410. Extraction consists of copying the resource information into a system resource file 1414. System resource 1414 is then polarized at resource polarization 1416 to become polarized system resource 1424. The polarization scheme for content polarization and resource polarization need not be the same. Preferably, each polarization scheme employs a polarization seed 1418 which is generated by seed generator 1426. Several exemplary methods for seed generation are described below. In particular, in a preferred embodiment, the polarization seed is based on unique information from the user's system.

Several techniques for generation of the polarization seed may be used. For example, a seed generator which generates a number from a random number generator may be used. This method, referred to as stateless polarization, does not depend on any secret key information and user system information. The process for stateless polarization yields a specific value for the system for polarization. The inherent vulnerability for digital security systems may be found in mishandling secret information,

mathematical complexity, and algorithmic complexity. Eliminating the secret information seals off one target of attack. With stateless polarization, a random number generator produces the polarization seed. In this case, once the polarization process is complete the seed is discarded without a trace. Hence, the security of the system is free from attack focused on compromising the secret information, and the user need not divulge sensitive information that may be deemed a privacy violation.

Another seed generator that may be used is a state-based generator. The state-based seed generator constructs a seed by first acquiring system state information from the user's replay system or rendering device. System state information includes hardware identifiers, system settings and other system state-related information. While there is much value in stateless polarization, other security requirements may require use of an inseparable link to a particular user system or device. By generating the polarization seed from system/device-specific information, the polarization engine will produce a digital work that is polarized to a form that corresponds to a specific system/device.

The polarization seed generator can also be tied to an authorization process. In authorization-based polarization, the seed generation can be tie in with the outcome of the authorization process. A separate authorization repository (which is a trusted source) provide authorization information as part of some other security feature associated with delivering access to a digital work to a user. The trusted source of authorization information may be an online authorization repository as described in US Patent No. 5,629,980. This authorization information is then used to generate a polarization seed.

If a stateless polarization seed is used, the digital work and its resource information may be polarized and stored together for delivery to a user when a user purchases the associated rights of use for the particular digital work. If one of the other polarization seed generation methods is used, polarization typically must wait until the user provides the system state or authorization information before the digital work and resource information may be polarized.

An embodiment which provides a higher level of protection in terms of ensuring that the digital work may be replayed only on a specific physical system or device uses a

dynamic state based polarization seed. In this embodiment, a polarization engine and polarization seed generator must be provided to the replay application or rendering device along with the digital work and resource information. In this embodiment, the digital work and resource information are polarized prior to replay and rendering using a seed which is generated based on the dynamic state of the particular system or device. The dynamic state may come, for example, from the system clock, CPU utilization, hard drive allocation, cursor coordinates, etc. By polarizing the work using a snapshot of a dynamic state, the work is locked to a particular system configuration (i.e., state) in time. Polarization of the digital work, and ultimately its blind replay (described below), is based upon a dynamically evolving state. The evolution of the dynamic state does not yield unique secret information that allows repeatability of the polarization process, and hence dynamic-state based polarization makes compromising the polarized digital work and system context more difficult. Since the polarization process is carried out within a trusted system, it is implied that the process can not be deconstructed.

The actual process of polarization can be, as described in the example above, an algorithmic-based transformation -parameterized by the polarization seed. During polarization, the data and resource identifiers of the digital work are transformed as described above. The structure of the digital work is unaltered, however, such that the original format, such as PDF, DOC, WAV, or other format, is retained much like in the format preserving encryption. Similarly the polarization of the resource information yields a polarized form of the resource information such that the resource identifiers, element identifiers and resource characteristics are transformed, yet the structure of the system context remains unaltered. By polarizing the digital work and resource information according to the same seed based on a user's specific device or system information, an inseparable relationship is established such that the work cannot be replayed to its clear form with any other device or user system. If circulated in an unauthorized manner, the protection remains in effect.

During blind replay, the unique characteristics of the polarized resource information enable the replay application to properly replay the polarized digital work

and generate unpolarized or clear presentation data. Because the digital work and the resource information were transformed in a complementary manner, the polarized elements of the digital work, such as the resource identifiers and data, unknowingly reference the complementary elements within the resources of the system context. Due to the matching transformation the proper elements within the context are identified by the replay application such that the resultant presentation data appears in the clear. Hence, the work is protected until the last possible moment after replay.

As discussed earlier, the conventional distribution of digital works via the web is relatively straightforward. The work is created using an editor, posted to a web site, accessed by the user audience and replayed in a viewer or on a display system. If a content owner does not desire to protect his/her digital work (or if the content owner trusts all users who will receive the work), the digital work is provided "in the clear" i.e., without any encoding, encryption or other protection for direct use by any user.

If the digital work is downloaded onto the user's system, it is typically stored in memory. If the digital work is provided via a storage media, such as floppy disk or CD-ROM or DVD-ROM, the digital work is usually accessed directly from storage media.

In order to play the digital work, referring to Figure 15, the digital work 1510 is provided to a replay application 1512. In the case of a document or other type of digital work which requires formatting information or resource information, the digital work will include digital content plus resource information setting forth the particular system context or system resources needed by the replay application to process the digital content. For example, the digital work 1510 may be a text document in which the text is displayed using the Arial font. When replay application 1512 accesses resource information on digital work 1510 indicating Arial font is used, it accesses the appropriate system resources 1516 (which in this case is the Arial font table) and uses the system resource information to convert the digital content into presentation data 1514.

In some replay applications, converting the digital content into presentation data is sufficient for use by the user. In others, presentation data is only an intermediate form which must be further converted. For example, in the case of a display system 1524

which is a printer, the presentation data 1514 must be further rendered by rendering application 1518. Rendering application 1518 may be a decomposer within the printer. Rendering application 1518 uses other system resources 1516 to transform the presentation data 1514 into image data 1520. Image data 1520 is in a form which can be directly displayed on display device 1522 (in the case of a printer, output as a printed document).

In addition to the earlier described systems and methods for protecting a digital work during replay, a digital work may be protected during replay by polarizing the digital work in accordance with a first polarization scheme which produces polarized content and preserves the digital work's resource information. A portion of the digital work's resource information is copied and polarized in accordance with a second polarization scheme. Referring to Figure 16, replay application 1612 uses the polarized resource information 1614 (and any other system resource information 1616 that may be required) to transform the polarized digital work1610 into clear presentation data 1618. Presentation data is necessarily in the clear, which means it can be captured by other programs (such as a screen capture utility program). However, the output of such other programs is not in the same format and frequently not of the same fidelity as the original digital work.

The polarized resource information can be thought of as acting like a polarizing filter to bring the polarized digital content into a clear image (presentation data). This system is a blind replay system in that the replay application, which can be any commercial application, does not know or need to know the clear digital content. Blind replay operates for any transformation function R, such that $R(w',s') = R(w,s)$, where w' is the polarized digital content, w is the clear digital content, s' is the polarized resource information and s is the unpolarized resource information. Blind replay of polarized digital works using polarized resource information is different from blind transformation described above in that blind replay produces clear presentation data without having to depolarize it. In blind transformation, the replay application converts the encrypted

digital work into encrypted presentation data, which must then be decrypted. In both cases, the user does not see the original digital work in clear form.

Blind replay (also called blind rendering) using a polarized digital work and polarized resource information can be used alone to protect the digital work during replay as well as in addition to regular encryption. For example, the polarized digital work and polarized resource information may be encrypted to protect it during distribution, then decrypted at the user's system into the polarized digital work and polarized resource information. The user must first obtain permission from the content owner or the distributor acting on behalf of the content owner (in order to decrypt the encrypted digital work). Once the user is qualified, the encrypted polarized digital work and the encrypted polarized resource information are decrypted and the polarized digital work is replayed in the replay application using the polarized resource information.

The complexity of rendering a digital work into a usable form for viewing by a user can be used to further protect the digital work during replay. Referring to Figure 17, polarized digital work 1710 is provided to replay application 1712, which uses polarized system resources 1716 and other system resources 1718 to transform polarized digital work 1710 into partially polarized presentation data 1714. In this embodiment, display system 1728 is needed to transform presentation data into a form usable by the user. Partially polarized presentation data 1714 is provided to rendering application 1720 which uses polarized system resources 1716, local system resources 1722 and system resources 1718 to transform the partially polarized presentation data 1714 into clear image data 1724. Clear image data 1724 is then displayed on display device 1726 for use by the user. In this embodiment, presentation data is still polarized, taking the location of the clear data to a later point of the display process and providing further protection.

To enhance usability of the system for polarization of digital works, the polarized resource information may be separated from the digital work and tied to a transportable device such as a smart card. In this embodiment, the replay application 1712 plays back the work using the polarized system resources 1716. Instead of having the polarized system resources 1716 stored in a local memory, along with the polarized digital work,

1710, the polarized system resources 1716 is stored in a transportable device such as a smart card. Also, the smart card, possibly with hardware-enhanced features, may possess attributes that provide for tamper resistance. Within the transportable context, the polarized data is processed by the replay application 1712 to yield the partially polarized presentation data and then provided to the rendering application 1720.

Many different types of digital works can be protected throughout use using the polarization method. For example, if the digital work is a document or text file, the replay application may be a word processor, system resources or resource information may include font tables, page layout, and color tables. If the digital work is audio or video data (e.g., streams), the replay application may be an audio or video player. The presentation data will be the audio/video final data stream. The display system may be an audio/video device. The rendering application may be the audio/video device driver. The image data may be the audio/video device data stream and the display device may be the audio/video rendering device (speaker or monitor, for example).

For a digital work that is an audio/video data stream, the system resources or resource information may include characteristics of the audio/video device: sample rate (samples per second – e.g., 8 kHz, 44.1kHz), sample quality (bits per sample – e.g., 8, 16); sample type (number of channels – e.g., 1 for mono, 2 for stereo), and sample format (instructions and data blocks). A table of some audio/video data streams and their corresponding resource information or variable parameters which can be selected for polarization is set forth below:

Extension	Origin	Variable Parameters (#Fixed)	Compression	Player
.mp3	MPEG standard	sample rate, quality, #type	MPEG	MP3 Player
.ra	Real Networks	sample rate, quality, #type	Plug-ins	Real Player
.wav	Microsoft	sample rate,	ADPCM	Window Media

		quality, #type		
.snd	Apple	sample rate, #quality, #type	MACE	QuickTime

Table 1: Digital Work: A/V Data (Streams)

The structure of a digital work can be used advantageously for polarization. While it is possible to polarize the entire digital work, it is more convenient to polarize only a portion of the digital work. Most digital works include three primary elements: instructions, data, and resources. Preferably, only the data and resources of the digital work are polarized, much like the format preserving encryption method described above. By selectively transforming only the data and resources, a digital work may be transformed such that the content remains in the original format, yet the data and resources are incomprehensible.

The general layout of a digital work of the document type is shown in Figure 18. In Figure 18, digital work 150 includes Page Descriptor 152, Control Codes 154, 158 and 162, Resource Identifier 156, and Data 160 and 164. The Page Descriptors 152 define the general layout of a work. For instance, the page size, page number, and margins fall into the category of Page Descriptors with respect to digital documents. Control Codes 154, 158 and 162 are similar in that they describe the presentation of the content. Examples include commands to set text position, output text, set font type, and set current screen coordinates. Resource Identifiers 156 simply reference the desired resources. In the digital document realm, resources could vary from font typeface to background color. Finally, Data 160, 164 represent the core information communicated by the digital work. This could be the drawing coordinates used in a multimedia clip or the character codes for rendering as a digital document.

An example of a digital work (in this case a simple digital document) and one of its polarized forms are shown in Figures 19 and 20, an HTML document in clear and polarized form. The tags <html> and <body> are Page Descriptors. The <font>...<\font> tag is an example of a Control Code for setting font resource

characteristics, while "Arial" and "14" are Resource Identifiers for an Arial typeface, 14 point font. The "Hello World" text is the Data, or the core information of the work. The <p> is another Control Code to signal the beginning of the paragraph. Finally, the document is closed out with Page Descriptors <\body> and <\html> to identify the end of the document.

Figure 20 shows what the digital work of Figure 19 looks like in a polarized form. It can be seen that the Page Descriptor and Control Code tags remain unaltered; the <html>, <body> and <font> tags are unchanged. Whereas, the Resource Identifiers, "Arial" and "14", have been transformed to indecipherable values. Similarly, the Data, "Hello World", has also been transformed to an indecipherable value. By transforming the Resource Identifiers and the Data the content is rendered meaningless while in the polarized form. Yet, the fact that the Page Descriptors and Control Codes remain intact allows for the document to retain its original format, which in general could be HTML, Adobe PDF, RealNetworks RAM, Apple QuickTime, etc.

The system context (or system resources or resource information) can be thought of as the collection of system resources available to a replay application on a particular system. For example, it may include the Font Table, Color Palette, System Coordinates and Volume Setting. When a digital work is input to a replay application, the replay application uses the particular resource information contained within the digital work to transform the digital content into presentation data. Each system context or resource information contained within a digital work is or can be altered to be unique to a system for which it can be replayed. The system context is a required element for the use of the digital work, tying use of the digital work to a specific system or physical device or replay application for replay. The Resource Identifiers and Data within the digital work may either directly or indirectly reference elements contained within the system context. Polarizing the digital work and system context enable blind rendering into clear presentation data. By polarizing the system context with a polarization seed that is tied to a unique system, the resulting polarized system context can be a unique environment in

which a complementary polarized digital work, which has been polarized with the same polarization seed, may be accessed and replayed.

Figure 21 illustrates a typical configuration of the system context. The elements include the resource identifier (ResID), element identifier (ElemID), and resource characteristics (Characteristics). The ResID includes pertinent information for other system components to reference the resources. The ElemID is the identifier of an individual element within the resource. Finally, the Characteristics are the actual resource characteristics used to express the individual resource element.

Figure 22 is an illustration of the resource for the font table pertaining to the Arial typeface. The key resource identifier in this case is the font name, "Arial". Following the ASCII convention, the number 48 identifies the individual resource element identifier. The resource element characteristics for the ElemID represent the information to express the letter 'a'.

Figure 23 is an illustration of the polarized the system context for the font resource shown in Figure 22. The resource identifier itself is transformed to "k13k2". The element identifier itself need not be transformed, as it is sufficient enough to transform the resource characteristics alone. In this case, "48" is depicted as transformed to express the characteristics for 'Y' instead of 'a'.

Polarization and blind rendering may be used for many different types of digital works. In addition to documents, polarization and blind rendering may be used for audio/video data. As noted above, audio/video data is generally provided in the form of streams. A replay application is the audio/video player which transforms the digital audio/video stream into a final data stream which can be processed by a transducer (speaker) into an audio output or by a display into a video image.

Referring to Figure 17, replay application 1712 corresponds to an audio/video player which generally operates by sampling the audio/video input streams 1710 at some sample rate, quality and type accepted by a target audio/video device. It uses the audio/video system resources to sample, mix and produce audio/video streams and then mixes the resampled audio/video streams to produce a final audio/video stream in a

format expected by the target device. In the case of an audio/video player, the presentation data 1714 is the final mixed audio/video stream at some sample rate, quality, type and format expected by a target audio/video device.

The target audio/video device (e.g., rendering application 1720) is some hardware system that is able to convert the audio/video stream (presentation data 1714) at a specific sample rate, quality, type (channel) and format (e.g., PAL or NTSC) to the device audio/video data 1724. Examples of audio devices include sound cards, speakers, monitors and the digital to analog converter located within the audio/video device. Many devices are able to play audio/video streams at a range of different sample rates. Image data 1724 (e.g. an audio signal or a video image stream) is generated by the audio/video device driver 1720 and "consumed" by the display device 1726.

For example, to polarize an audio/video data stream, it may be split into two or more separate streams. One stream is polarized and one stream is unpolarized. Each stream may have different device characteristics (resource information): sample rates, channels, qualities and/or formats associated with it. The device characteristics (one or more of the stream's sample rates, channels, qualities and/or formats) may also be polarized to generate the polarized resource information.

Blind replay of the polarized audio/video stream is accomplished in a similar manner as for a polarized digital document. The replay application (audio/video player) mixes together the unpolarized stream and the polarized stream, and using the polarized resource information, produces a polarized final data stream for the target audio/video device with a correct set of resource information. The target device (1720) uses the polarized resource information to play the polarized data stream generating clear sound/visual effects (1724).

While certain exemplary embodiments of the invention have been described in detail above, it should be recognized that other forms, alternatives, modifications, versions and variations of the invention are equally operative and would be apparent to those skilled in the art. The disclosure is not intended to limit the invention to any particular embodiment, and is intended to embrace all such forms, alternatives,

modifications, versions and variations. For example, the portions of the invention described above that are described as software components could be implemented as hardware. Moreover, while certain functional blocks are described herein as separate and independent from each other, these functional blocks can be consolidated and performed on a single general-purpose computer, or further broken down into sub-functions as recognized in the art. Accordingly, the true scope of the invention is intended to cover all alternatives, modifications, and equivalents and should be determined with reference to the claims set forth below.

What is Claimed is:

1. A method of protecting a digital work, z, during transformation by a transformation function, F, into presentation data F(z), wherein the digital work includes digital content and formatting information, comprising:

encrypting the digital work, z, in accordance with a format preserving encryption scheme, E;

transforming the encrypted digital work E(z) into encrypted presentation data, F(E(z)); and

decrypting the encrypted presentation data, F (E(z)), in accordance with a decryption function, D, to obtain the presentation data, F(z), wherein D(F(E(z)) = F(z).

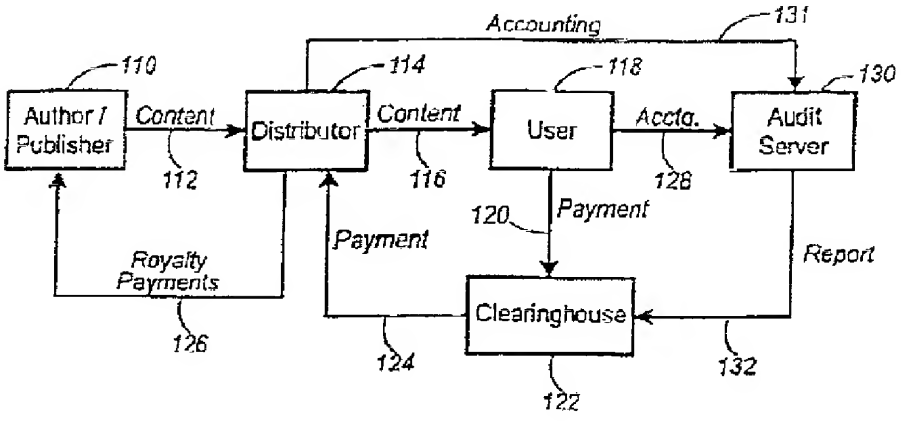
2. A system of protecting a digital work, z, during transformation by a transformation function, F, into presentation data F(z), wherein the digital work includes digital content and formatting information, comprising:

an encryption engine for encrypting the digital work, z, in accordance with a format preserving encryption scheme, E;

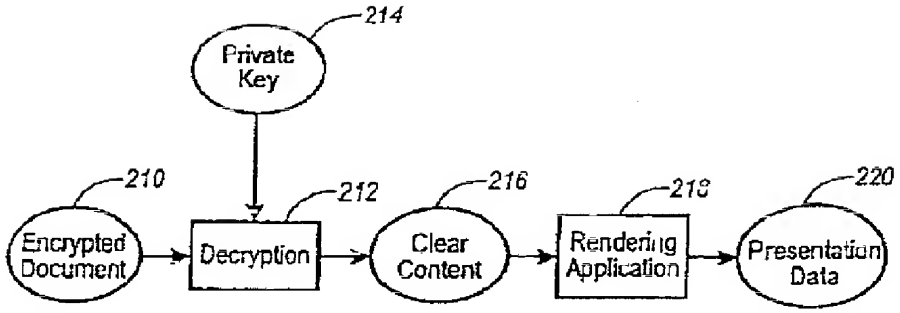
a transformation function for transforming the encrypted digital work E(z) into encrypted presentation data, F(E(z)); and

a decryption engine for decrypting the encrypted presentation data, F (E(z)), in accordance with a decryption function, D, to obtain the presentation data, F(z), wherein D(F(E(z)) = F(z).

【図１】

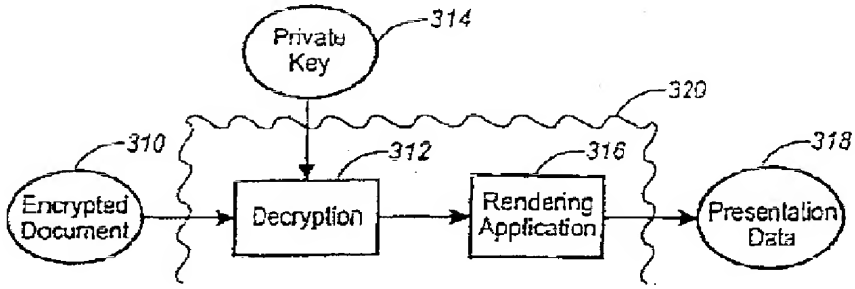


【図２】

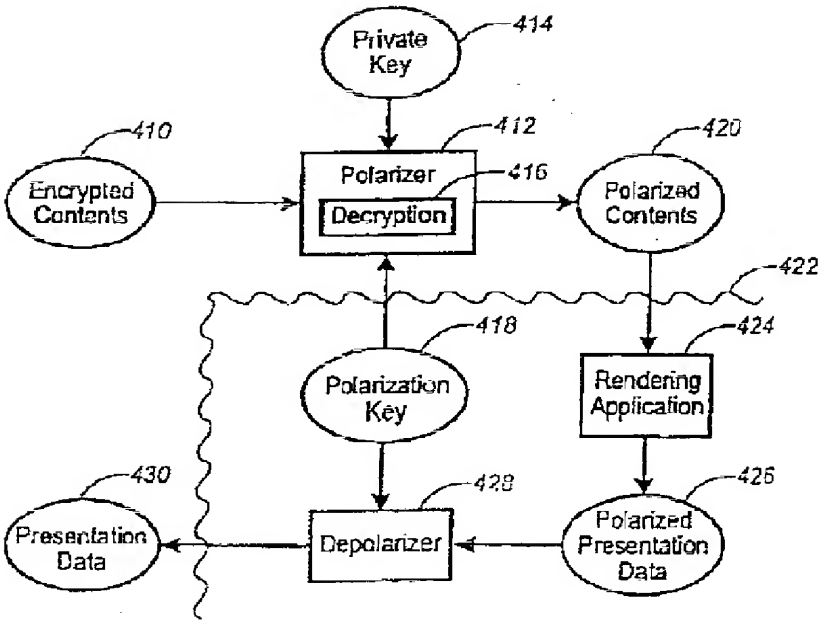


Prior Art

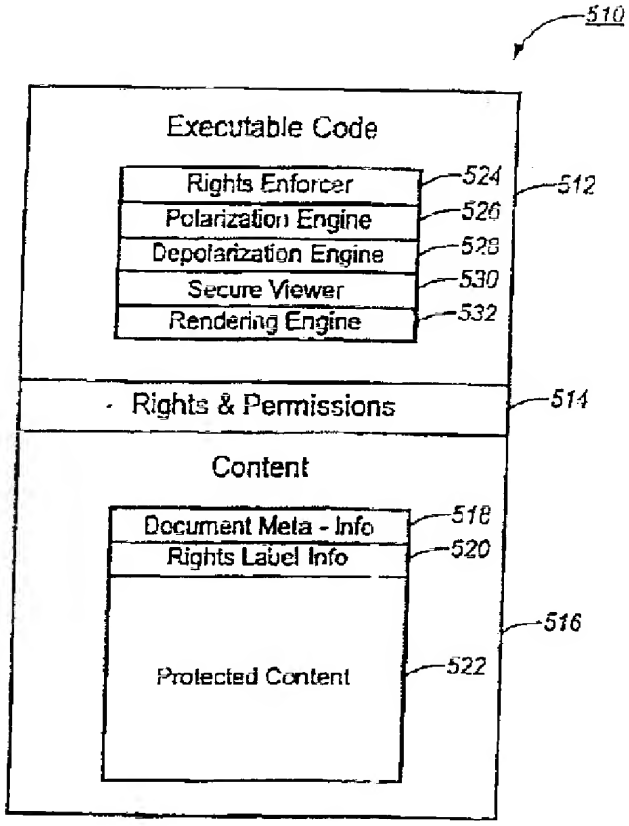
【図３】



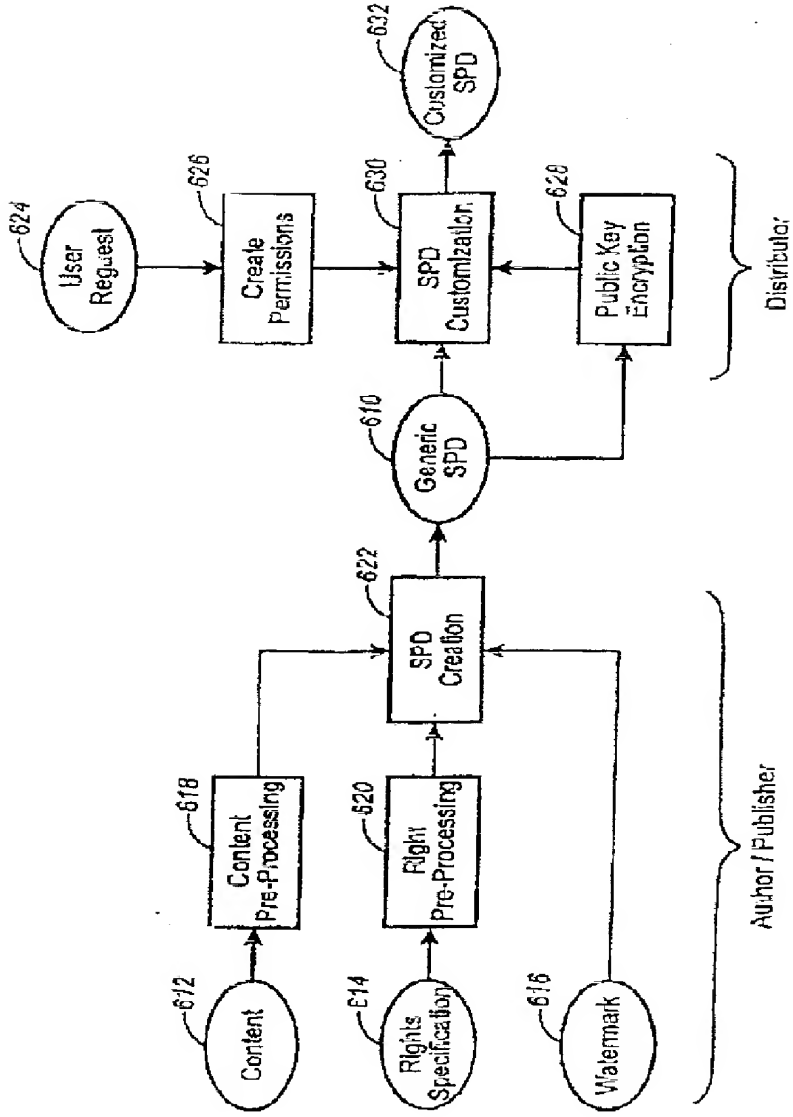
【図4】


Private Key
414
410
412
420
Encrypted Contents
Polarizer
416
Decryption
Polarized Contents
422
418
424
Polarization Key
Rendering Application
430
428
426
Presentation Data
Depolarizer
Polarized Presentation Data

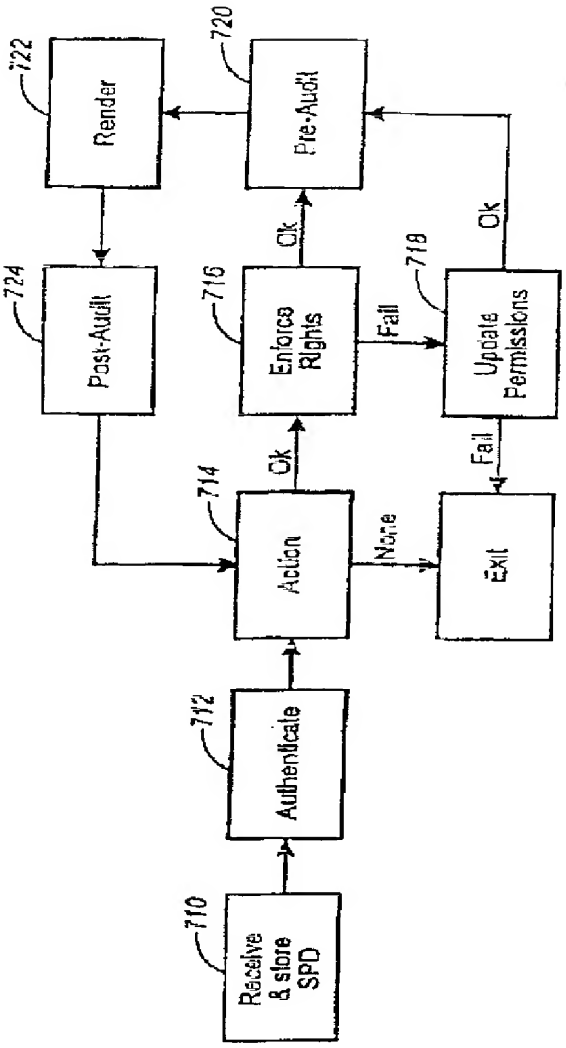
【図5】


510
Executable Code
Rights Enforcer
524
Polarization Engine
526
Depolarization Engine
528
Secure Viewer
530
Rendering Engine
532
512
Rights & Permissions
514
Content
Document Meta - Info
518
Rights Label Info
520
516
Protected Content
522

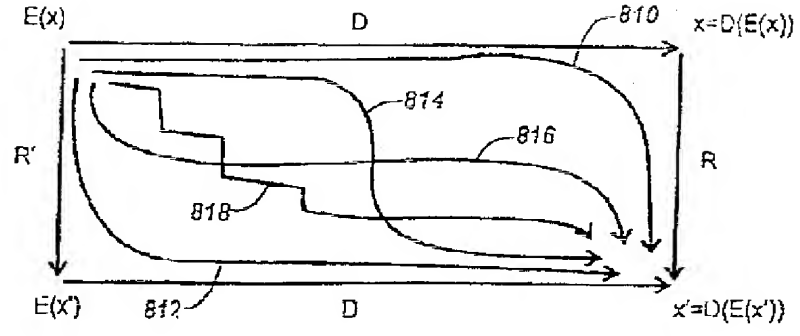
【図6】


User Request
624
Create Permissions
626
SPD Customization
630
Customized SPD
632
Public Key Encryption
628
Distributor
Generic SPD
610
SPD Creation
622
Content Pre-Processing
618
Right Pre-Processing
620
Content
612
Rights Specification
614
Watermark
616
Author / Publisher

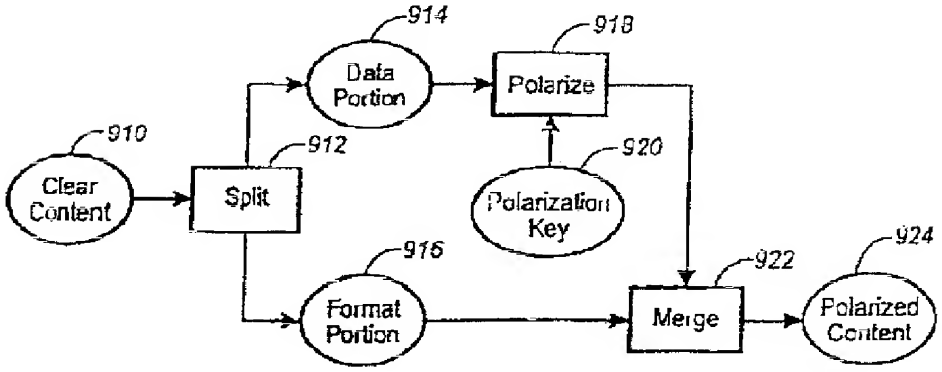
【図7】


710
Receive & store SPD
712
Authenticate
714
Action
716
Enforce Rights
718
Update Permissions
720
Pre-Audit
722
Render
724
Post-Audit
Ok
Fail
None
Exit

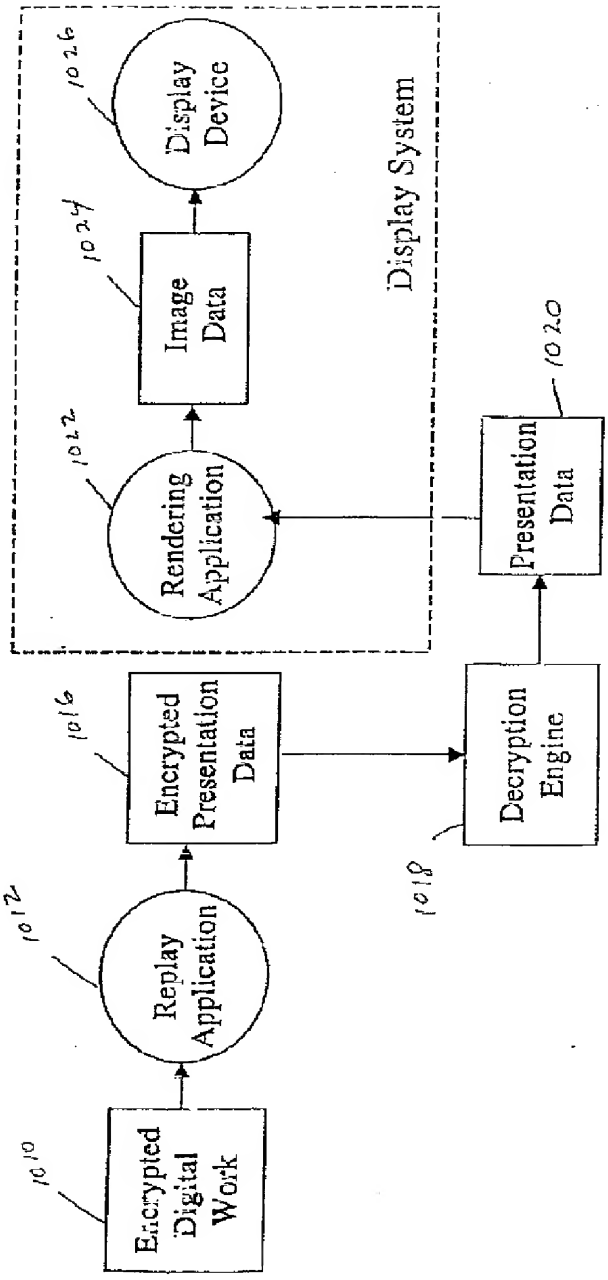
【図８】


E(x)
D
810
x=D(E(x))
814
816
R'
R
818
E(x')
812
D
x'=D(E(x'))

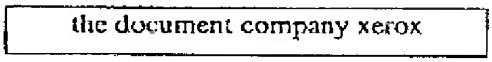
【図９】


914
Data Portion
918
Polarize
910
Clear Content
912
Split
920
Polarization Key
916
Format Portion
922
Merge
924
Polarized Content

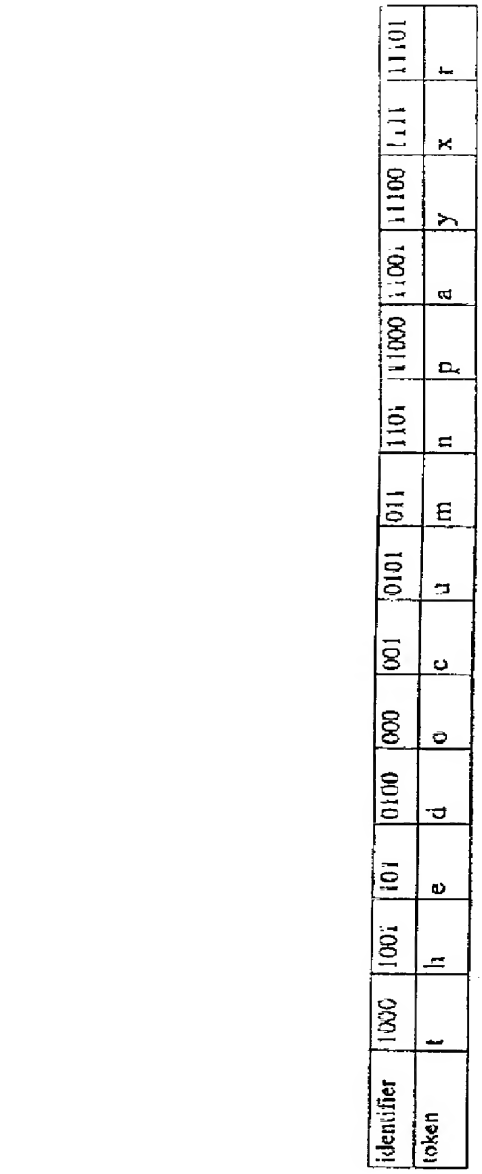
【図１０】


Display Device
1026
Image Data
1024
Rendering Application
1022
Display System
Presentation Data
1020
Encrypted Presentation Data
1016
Decryption Engine
1018
Replay Application
1012
Encrypted Digital Work
1010

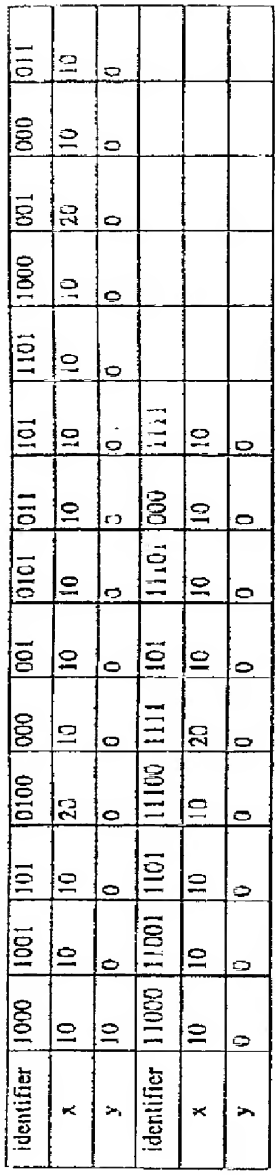
【図１１】


the document company xerox

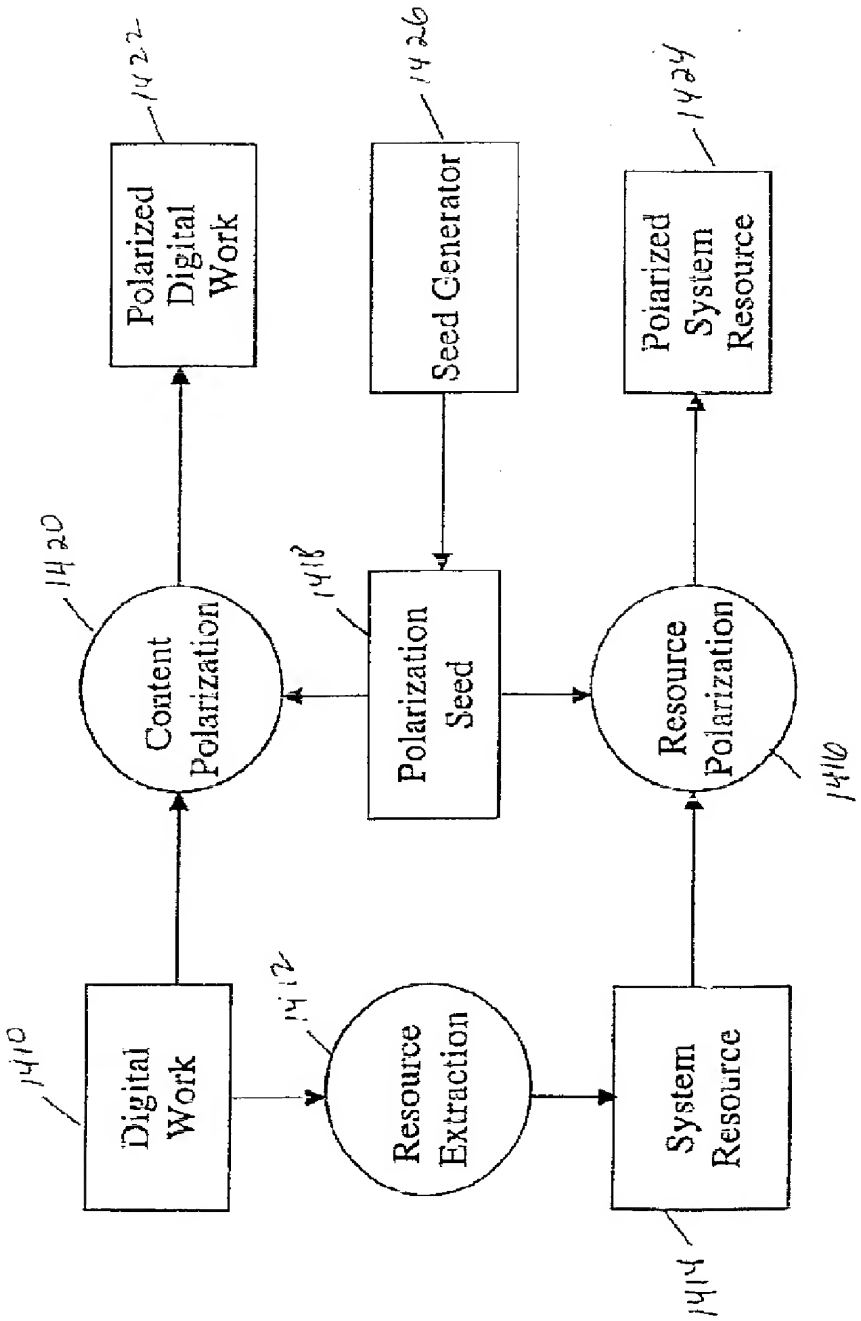
【図12】


identifier
token
10000
t
10001
h
101
e
0100
d
000
o
001
c
0101
u
011
m
11010
n
11000
p
11001
a
11100
y
1111
x
11101
r

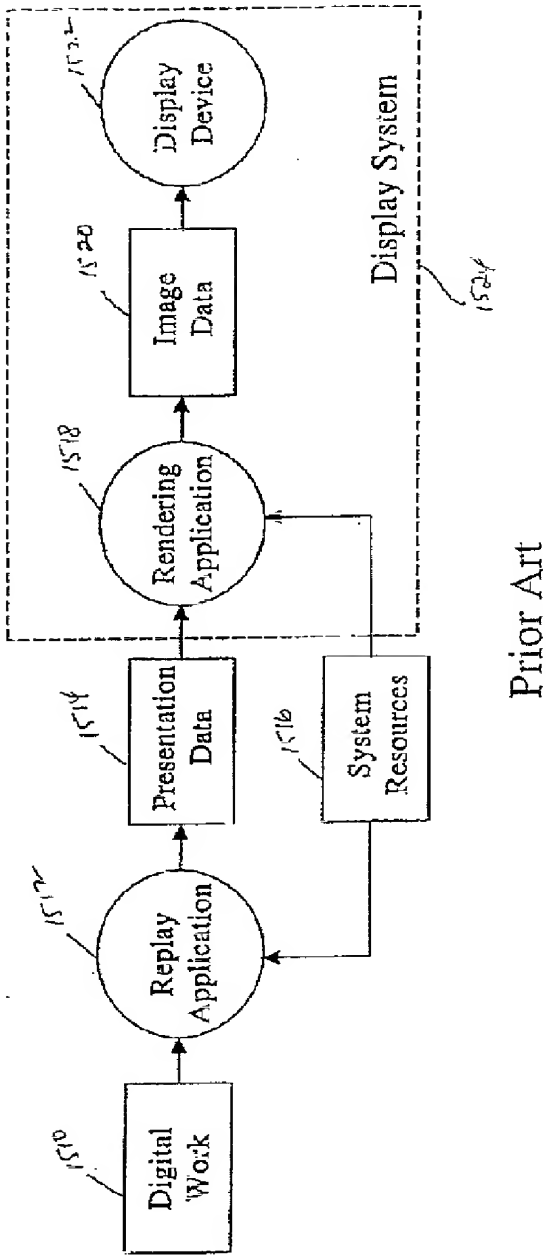
【図13】


identifier
1000
1001
101
0100
000
001
0101
011
101
1101
1000
001
000
011
x
10
10
10
25
10
10
10
10
10
10
25
10
10
y
10
0
0
0
0
0
0
0
0
0
0
0
0
0
identifier
11000
11001
1101
11100
1111
101
11101
000
1111
x
10
10
10
10
20
10
10
10
y
0
0
0
0
0
0
0
0

【図14】

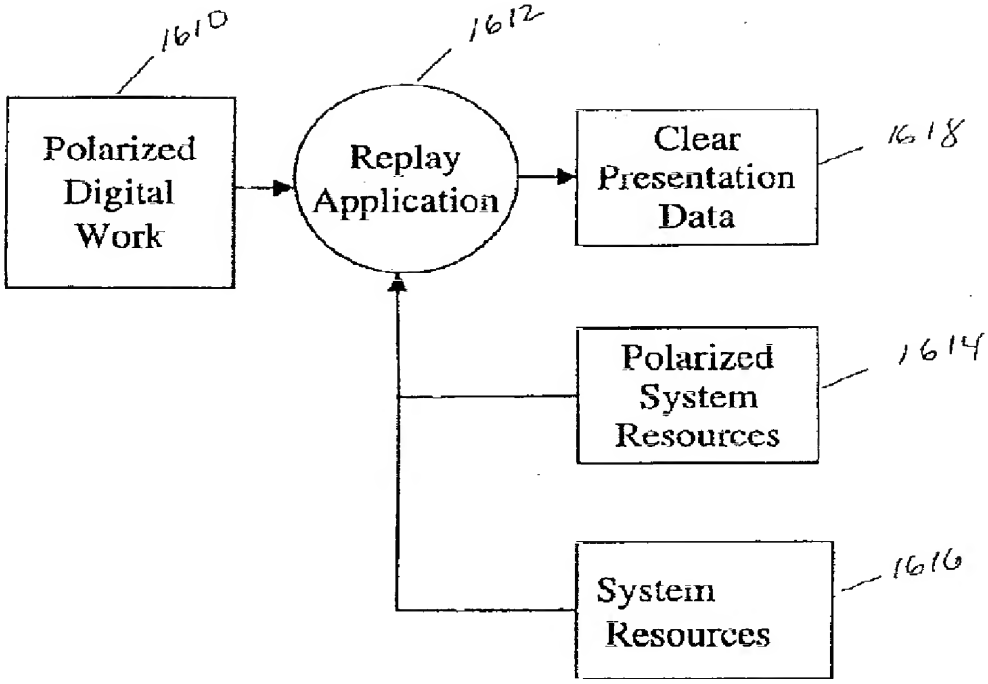

Digital Work
1410
Resource Extraction
1412
System Resource
1414
Content Polarization
1420
Polarization Seed
1418
Resource Polarization
1416
Polarized Digital Work
1422
Seed Generator
1426
Polarized System Resource
1424

【図１５】

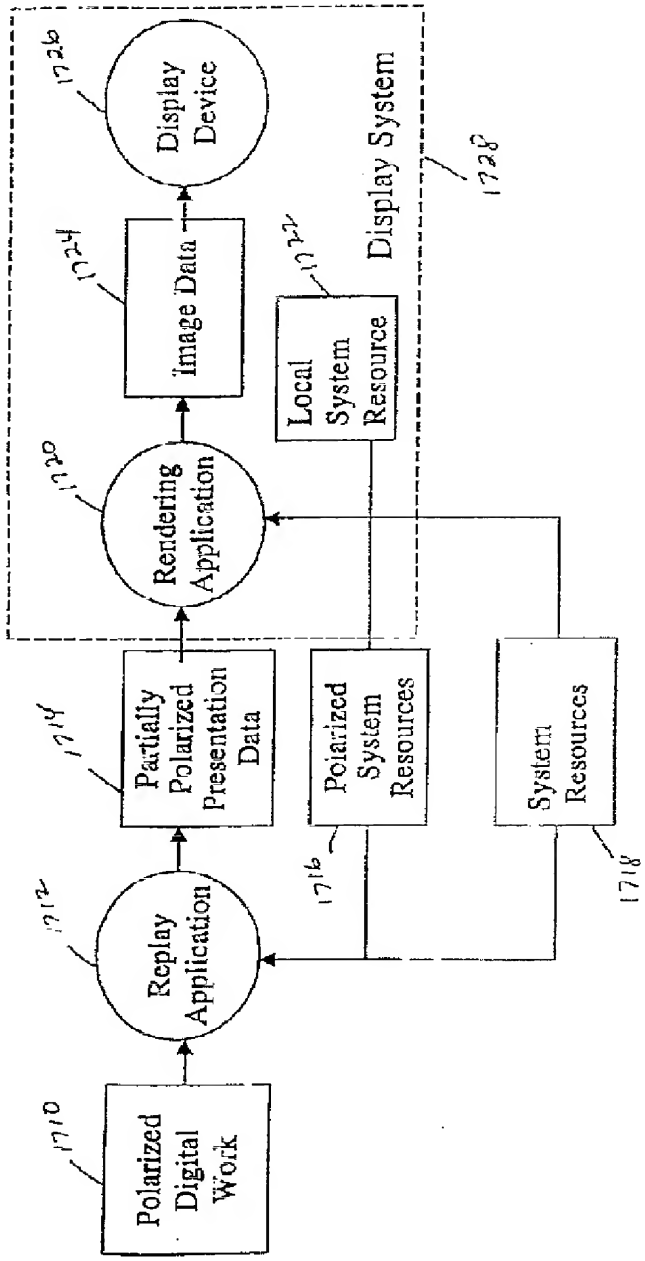


Prior Art

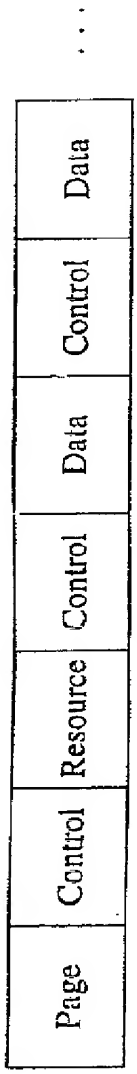
【図16】


1610
1612
Polarized Digital Work
Replay Application
Clear Presentation Data
1618
Polarized System Resources
1614
System Resources
1616

【図17】


Display Device
1726
Image Data
1724
Local System Resource
1722
Display System
1728
Rendering Application
1720
Partially Polarized Presentation Data
1714
Polarized System Resources
1716
System Resources
1718
Replay Application
1712
Polarized Digital Work
1710

【図18】



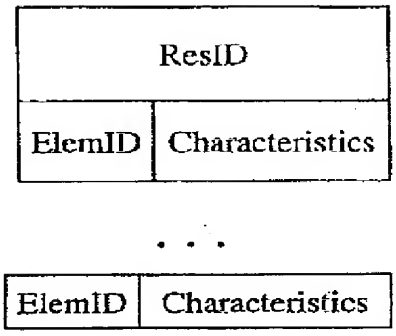
【図19】

```
<html>
<body>
<font name="Arial" size="14">
Hello World
</font>
<p>
</body>
</html>
```

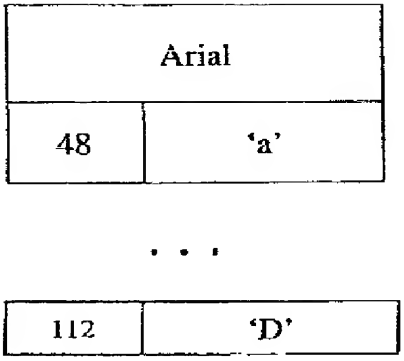
【図20】

```
<html>
<body>
<font name="kl3k2" size="21">
v0aa 8 aa0
</font>
<p>
</body>
</html>
```

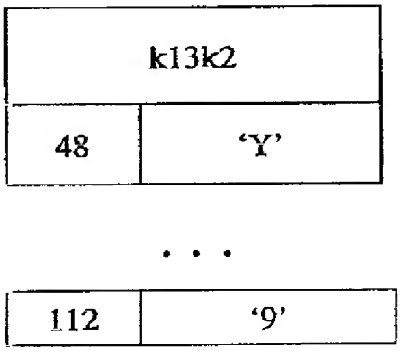
【図21】



【図22】



【図23】



Abstract of the Disclosure

A method of protecting a digital work uses a format preserving encryption scheme to encrypt the digital work. This enables any native replay application or rendering application to transform an encrypted digital work into encrypted presentation data. The originator's digital content is protected in its original form by not being decrypted. This method enables the rendering or replay application to process the encrypted document into encrypted presentation data without decrypting it first. Encrypted presentation data is then decrypted just before it is displayed to the user. An additive encryption scheme is a particular type of encryption scheme which preserves formatting of a digital work.